



**AUTOMATIC TARGET RECOGNITION
USER INTERFACE TOOL**

THESIS

David A. Kerns, Capt, USAF
AFIT/GOR/ENS/07-15

**DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY**

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED

The views expressed in this thesis are those of the author and do not reflect the official policy or position of the United States Air Force, Department of Defense, or the United States Government.

AFIT/GOR/ENS/07-15

AUTOMATIC TARGET RECOGNITION USER INTERFACE TOOL

THESIS

Presented to the Faculty

Department of Operational Sciences

Graduate School of Engineering and Management

Air Force Institute of Technology

Air University

Air Education and Training Command

In Partial Fulfillment of the Requirements for the

Degree of Master of Science in Operational Science

David A. Kerns, BS

Captain, USAF

March 2007

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED

AFIT/GOR/ENS/07-15

AUTOMATIC TARGET RECOGNITION USER INTERFACE TOOL

David A. Kerns, BS
Captain, USAF

Approved:

Dr. Kenneth W. Bauer (Chairman)

date

Dr. Mark E. Oxley (Member)

date

Abstract

A computer tool to aid in selecting the best Automatic Target Recognition (ATR) algorithm is developed. The program considers many quantifiers, accepts user-defined parameters, allows for changes in the operational environment and presents results in a meaningful way. It is written for Microsoft Excel.

An ATR algorithm assigns a class label to a recognized target. General designations can include “Friend” and “Foe.” The error of designating “Friend” as “Foe” as well as “Foe” as “Friend” comes with a high cost. Studying each algorithm’s error can minimize this cost. Receiver Operating Characteristic (ROC) curves provide only information on the probabilities given a system state of declaring up to three class labels: “True,” “False” or “Unknown.” Other quantifiers, including an alternate ROC curve, are developed in this study to provide information on the probability of a system state given any of multiple declarations, which is more useful to the user. Sensitivity to prior probabilities, suggestions for user-defined parameters and areas for future research are identified as the User Interface Tool is described in detail in this thesis.

AFIT/GOR/ENS/07-15

To my family and all the friends who make me feel like family.

Acknowledgements

I would like to express my sincere appreciation to my thesis advisor, Dr. Kenneth Bauer, for his direction throughout this thesis research. I would also like to thank my sponsor for his timely assistance and availability to answer any of my questions. I would like to thank AFIT captains Ryan Caulk, Kevin Reyes, Yuri Taitano, Joe Bellucci, and Jason Williams for being the very definition of a “Super Group.” I would like to thank other company grade officers – including Scott Elkins, Jennifer Elkins, Erin Johnson, Darin Findling, Tetsuo Kaieda, Christopher Gardner and Briana Smith – for their support in helping me make time for what is important to me and letting me know I am important to them.

Thank you to my loving parents and siblings for their understanding and support. Finally, I would like to thank God for giving me strength, guidance, friends and family to see me through the rough times. “Any fool can agree with you when you’re right. A true friend will agree with you when you’re wrong.”

David A. Kerns

Table of Contents

	Page
Abstract	iv
Acknowledgements.....	vi
Table of Contents.....	vii
List of Figures.....	x
List of Tables	xiii
 I. Introduction	 1
Background.....	1
Problem Statement	1
Input.....	2
Data File.....	3
Algorithms and Parameters.....	7
Optional Parameters	9
Output	10
ROC Curves	10
Summary Data.....	11
Cost Matrix	12
Research Objectives/Questions/Hypotheses.....	12
Research Focus	12
Methodology	13
Assumptions/Limitations.....	13
Implications	15
Preview	16
 II. Literature Review	 17
Historical Perspective.....	17
Error in Hypothesis Tests	17
Receiver Operating Characteristic	18

Error in ATR Comparisons.....	19
Feature Determination	21
Decision Determination.....	22
User-Defined Input Variables	24
Application of Historical Perspective.....	25
Possible Algorithm Performance Measures	26
Error Exploration	27
Error with “Unknown” Allowed	27
Error with Multiple Classifications.....	29
Decision-Making Process	32
Multiple Classifier Systems	32
Fusion Comparisons	34
III. Methodology	36
Input	36
Input Sheet Solutions.....	38
Master Worksheet	40
User-Defined Parameters.....	41
Algorithm Calculations	47
ROC Curves.....	53
ROC Curve Calculations	55
ROC Curve Points and Plots.....	61
Vertical ROC Curves.....	66
Calculating Vertical ROC Curves.....	68
Prior Uncertainty.....	72
ROC Curves Based on Normalized Thresholds.....	76
Summary Statistics.....	79
IV. Results and Analysis	84
First Analysis	86
Second Analysis.....	103
Conclusion	111
V. Discussion.....	113

Limitations.....	113
Future Studies	113
Confusers as “Unknown” rather than natural “Clutter”	114
Relevance of Unknown Misidentification Assumption	114
Use Solver to generate User-Defined parameters.	116
Other Future Research.	117
Contributions/Implications	117
Appendix A: Worksheet List.....	120
Appendix B: Pseudocode	122
Bibliography.....	133

List of Figures

Figure	Page
Figure 1. T2 (Tiger) Target Likeness Scores.....	5
Figure 2. T3 (Panzer) Target Likeness Scores.	5
Figure 3. T4 (T-72) Target Likeness Scores.	5
Figure 4. T5 (Bradley) Target Likeness Scores.....	6
Figure 5. T1 (Confuser or Clutter) Target Likeness Scores.	6
Figure 6. Receiver Operating Characteristic Curve comparing three algorithms.	19
Figure 7. Surface plot of ROC Curve. (Albrecht, 2005:98).....	29
Figure 8. Interface panel for opening a spreadsheet to be analyzed.	41
Figure 9. User-Defined Parameters Blank Sheet.	44
Figure 10. User-Defined Parameters Filled-in.	44
Figure 11. Placement of data for algorithm calculations.	48
Figure 12. Snapshot of the entire “Algorithms” worksheet.	49
Figure 13. Area for user to change user-defined parameters.....	49
Figure 14. Truth data calculations spreadsheet screenshot.	57
Figure 15. Positioning of Truth summary data.....	61
Figure 16. Bayes' Rule Applied.	63
Figure 17. ROC curve for one target.....	64
Figure 18. ROC curve with Current Thresholds.....	65
Figure 19. Placement for calculating Vertical ROC curves.	68
Figure 20. Vertical ROC Curve for Tiger.	69
Figure 21. Vertical ROC Curve for T-72.	70
Figure 22. User-Defined Priors.....	72

Figure	Page
Figure 23. Snapshot of a Priors worksheet.....	74
Figure 24. Priors Curve for Tiger.	74
Figure 25. Priors Curve for T-72.	75
Figure 26. Normalized ROC Curve for Tiger.....	78
Figure 27. Normalized ROC Curve for T-72.	78
Figure 28. Snapshot of "Stats" worksheet.....	79
Figure 29. Steps for computing a Cost Matrix.	83
Figure 30. Sample data parameters.....	84
Figure 31. Tiger scores showing True Scores range between 0.49 and 1.0.....	85
Figure 32. ROC Curve for Target 2, Tiger.....	87
Figure 33. ROC Curve for Target 3, Panzer.....	88
Figure 34. ROC Curve Target 4, T-72.	88
Figure 35. ROC Curve Target 5, Bradley.	89
Figure 36. Adjusted Thresholds: Left (High Threshold), Right (Low).	90
Figure 37. D-Bound Current point changing with Delta-threshold.	92
Figure 38. Vertical ROC Curve for Target 2, Tiger.	93
Figure 39. Vertical ROC Curve for Target 3, Panzer.	93
Figure 40. Vertical ROC Curve for Target 4, T-72.	94
Figure 41. Vertical ROC Curve for Target 5, Bradley.....	94
Figure 42. Normalized ROC Curve for Target 2, Tiger.....	97
Figure 43. Normalized ROC Curve for Target 3, Panzer.....	97
Figure 44. Normalized ROC Curve for Target 4, T-72.....	97
Figure 45. Normalized ROC Curve for Target 5, Bradley.....	98

Figure	Page
Figure 46. Priors Curve for T2, Tiger, and A2, TLS-Bound.....	99
Figure 47. Priors Curve for T3, Panzer, and A2, TLS-Bound.....	99
Figure 48. Priors Curve for T4, T-72, and A2, TLS-Bound.....	100
Figure 49. Priors Curve for T5, Bradley, and A2, TLS-Bound.....	100
Figure 50. ROC Curve changes for Target 2, Tiger.	105
Figure 51. ROC Curve for Target 3, Panzer.....	105
Figure 52. ROC Curve Changes for Target 4, T-72.	105
Figure 53. ROC Curve Changes for Target 5, Bradley.....	106
Figure 54. First Analysis ROC Curve for Target 2, Tiger.	106
Figure 55. Second Analysis ROC Curve for Target 2, Tiger.	107
Figure 56. First Analysis Vertical ROC Curve for Target 3, Panzer.....	107
Figure 57. Vertical Second Analysis ROC Curve for Target 3, Panzer.....	108
Figure 58. First Analysis Vertical ROC Curve for Target 5, Bradley.	108
Figure 59. Vertical Second Analysis ROC Curve for Target 5, Bradley.....	109
Figure 60. Vertical ROC Curve for Panzer and D-Bound with New to the Right. ...	109

List of Tables

Table	Page
Table 1. Four runs from the sample data file.....	2
Table 2. Classifier Evaluation Quantifiers.	12
Table 3. Four runs from the sample data file repeated from Table 1.....	22
Table 4. Classifier Evaluation Quantifiers.	27
Table 5. Cost Confusion Matrix.	31
Table 6. Four runs from the sample data.....	36
Table 7. Sample Scores (Features) with invalid entries.....	37
Table 8. Algorithm responses to the run data for a set of parameters.	43
Table 9. An example run for TLS-thresholds.....	45
Table 10. Second TLS-threshold case.....	45
Table 11. Case 3 of TLS-Thresholds.	46
Table 12. Test Run #420 results.	46
Table 13. Run #420 with TLS-thresholds altered and Delta Scores calculated.	47
Table 14. Biggest-TLS Calculation.	51
Table 15. Biggest-D Calculation.	51
Table 16. “TLS & D” Calculations.....	52
Table 17. Truth data calculations.....	57
Table 18. Probability Matrix.	58
Table 19. Example for Alt_D.	59
Table 20. Table of Algorithms for one target.....	63
Table 21. Table of Algorithms for one target with Current Thresholds.	65
Table 22. Vertical ROC Curve Table for Tiger.....	69

Table	Page
Table 23. Count for Biggest-TLS.	80
Table 24. Recall Table.	80
Table 25. Accuracy Table.	81
Table 26. Precision and F-Score Table.	81
Table 27. Mutual Information Table.	82
Table 28. Classifier Evaluation Quantifiers.	82
Table 29. Summary Statistics for first analysis.	102
Table 30. First Analysis summary statistics.	110
Table 31. Second Analysis summary statistics.	110

I. Introduction

Background

Selecting the best algorithm for Automatic Target Recognition (ATR) can save lives by providing Air Force pilots with the best technique for distinguishing hostile targets from friendly targets in a given operational environment. Unfortunately, no standard measure is available to compare different ATR algorithms. Measures must be easy to interpret and allow for meaningful comparisons between each algorithm for the decision-maker to select the best algorithm. To meet this task, measures are identified and methods of presentations, like a Receiver Operating Characteristic (ROC) curve, are studied and in some cases improved. The sponsor requested a User Interface Tool (UIT) to allow for visual comparisons and to analyze costs and benefits between each algorithm. The tool must allow for changes in the operational environment. This thesis describes the creation of such a tool to meet the needs of the decision-makers and operators for an ATR UIT.

Problem Statement

Selection between ATR algorithms must provide useful information for the operator, provide insight for the programmer, and meet the demands of a changing operational environment. To meet these demands, the tool must compare five existing ATR algorithms provided by the customer, be able to use input data in the format provided, use Microsoft Excel as the primary software tool for mass availability, produce graphical information for quick comparisons between each algorithm, provide meaningful data for both the algorithm developer and the operator, and allow for

changes in the operational environment. The problem is more clearly identified after careful examination of what input is available and what output is expected of the tool.

Input

The customer requested both Input and Output to be accomplished through Microsoft Excel. The original input consisted solely of a data file listing test runs and the likeness of a target to an In-Library target. The input file consisted of a heading and the information return from a number of runs. Each information return was listed in a row. The row starts with a cell identifying the run. Next, the actual target recognized during the run is placed in the “Actual” column. The final five cells show a score between zero and one. The score shows how like the recognized target, identified in the “Actual” column, is to the target in the column heading of the cell. The heading and four rows of data are provided below in Table 1.

Table 1. Four runs from the sample data file.

	A	B	C	D	E	F	G
1	File_Name	Actual	T1	T2	T3	T4	T5
2	5_D1_1536_2048	Con1	0.513	0.517	0.525	0.393	0.601
3	5_D1_1536_2048	T2	0.48	0.596	0.421	0.372	0.457
4	5_D1_1536_2048	Con2	0.691	0.467	0.528	0.371	0.512
5	5_D1_1536_2048	Con3	0.632	0.651	0.519	0.502	0.53

In addition, user-defined parameters, target designations, and optional inputs may be available. User-defined parameters are thresholds required to solve each algorithm. Target designations are identifications of In-Library targets as more general classes of “Friend,” “Hostile,” or “Other.” Optional input may include the population estimates of all possible targets or costs for misidentifying targets.

Algorithm solutions are not provided. Instead, five existing algorithms for choosing a target were suggested. Necessary parameters to compute the solution for

each algorithm include thresholds of minimum values and thresholds of minimum Delta values. These thresholds were not provided. Algorithm solutions must be solved through the UIT.

The customer-requested input was further defined and studied before any programming could begin.

Data File

The test data file is a list of test runs comparing an actual target to scores against each item in the current library. The seven columns in the file included: one column for the details of the run, one for the name of the actual target, and five scores for each of the five items in the data file. The five items in library are named: T1, T2, T3, T4, and T5. For simplicity, these codes are replaced with more meaningful names in following sections. The names are Clutter, Tiger, Panzer, T-72 and Bradley, respectively.

Although given meaningful names, the targets in the test file are generic. They could pertain to any class – tank, aircraft, or Surface-to-Air Missile (SAM) site – and they can pertain to any classification: Hostile, Friendly, or Neutral. It is considered that the first target, T1, is not any object in particular, but is actually a “fuzz ball” target meant to identify Clutter – trees, bushes, or animals – and take them out of consideration from the other targets (Sadowski, 2006).

The scores correspond to an estimate of target probability or how “like” a detected object is to an object in the library (Parker, 2006). They cannot be broken down into individual assessments of likelihood, thus limiting the amount of fusion techniques possible for this study. They will be referred to as Target Likeness Scores, TLS, to prevent confusion with other scores.

In addition, these scores are not to be confused with likelihoods. They do not give a probability that the considered object is actually the target in the library. These scores do not sum to one. Each score provides a relative value of how “like” the two targets are, but some algorithms require parameters or “minimum thresholds” to be met before an object is considered for declaration as the In-Library target. Each algorithm uses these scores and parameters, but how they make use of them will be identified later in this study. It is not uncommon to have high scores rating a target similar to every other item in the library.

To illustrate how scores may be related to one another, incorrect scores can be plotted against correct scores for each target. These ordered scatter plots are included as Figure 1 through Figure 5. Each graph highlights a particular In-Library target. All runs having the highlighted target as the recognized, actual target are selected for that graph. The scores for incorrect targets are plotted on the y-axis against the score for the true target on the x-axis. Thus, all points plotted above the true score line are bad as they indicate an incorrect target received a higher Target Likeness Score, so the detection system found the object more like an incorrect target. Scores below the true score line are good, since these indicate the detection system gave the higher Target Likeness Score to the true target.

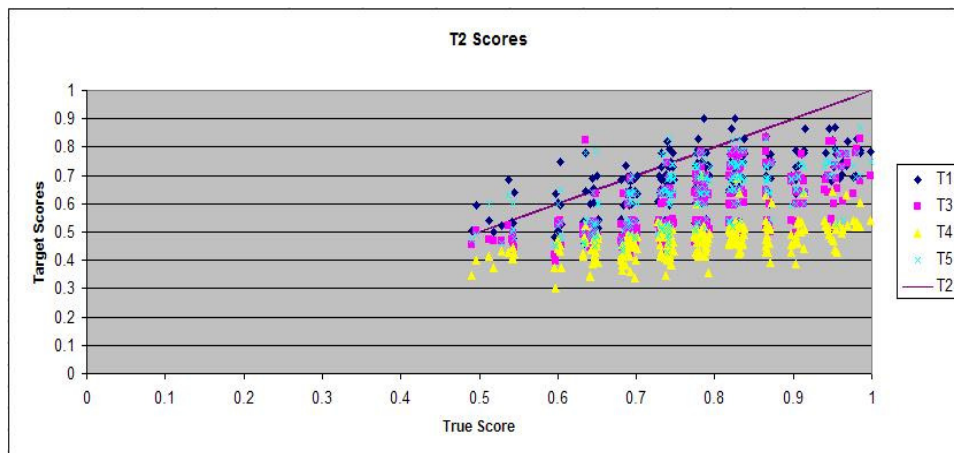


Figure 1. T2 (Tiger) Target Likeness Scores.

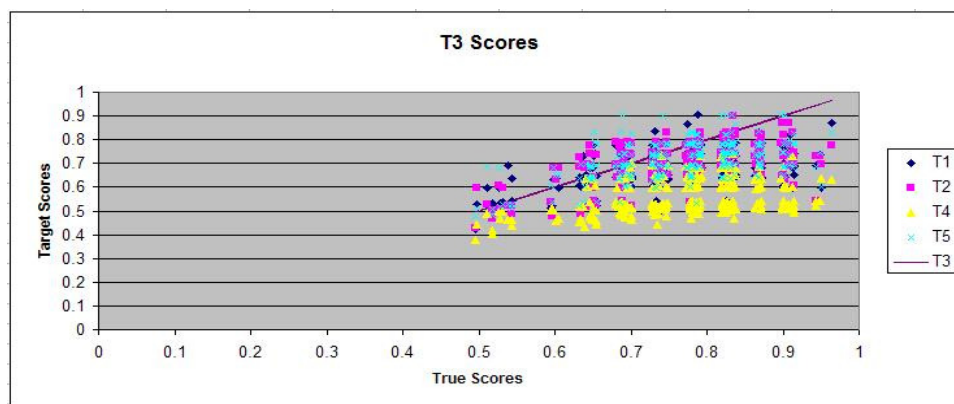


Figure 2. T3 (Panzer) Target Likeness Scores.

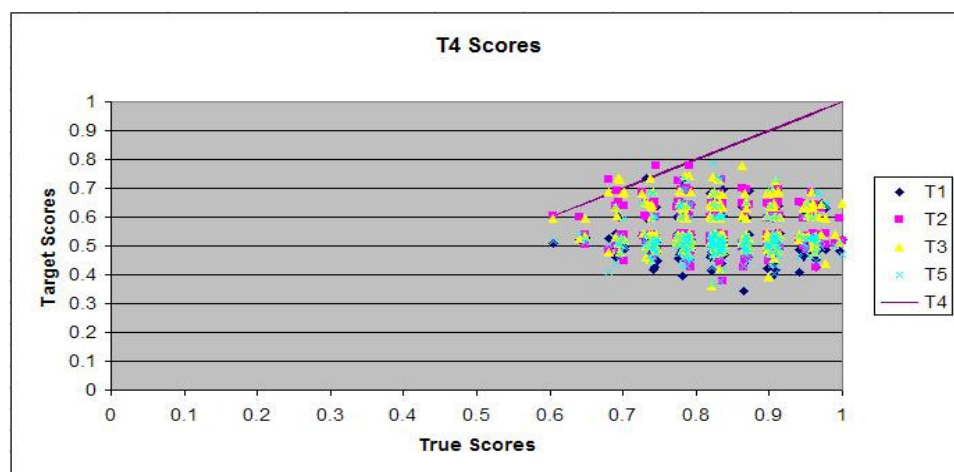


Figure 3. T4 (T-72) Target Likeness Scores.

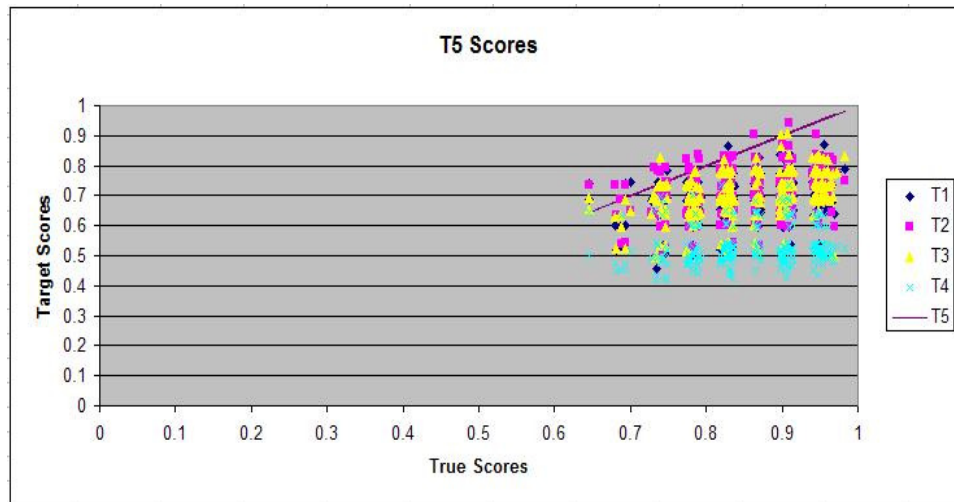


Figure 4. T5 (Bradley) Target Likeness Scores.

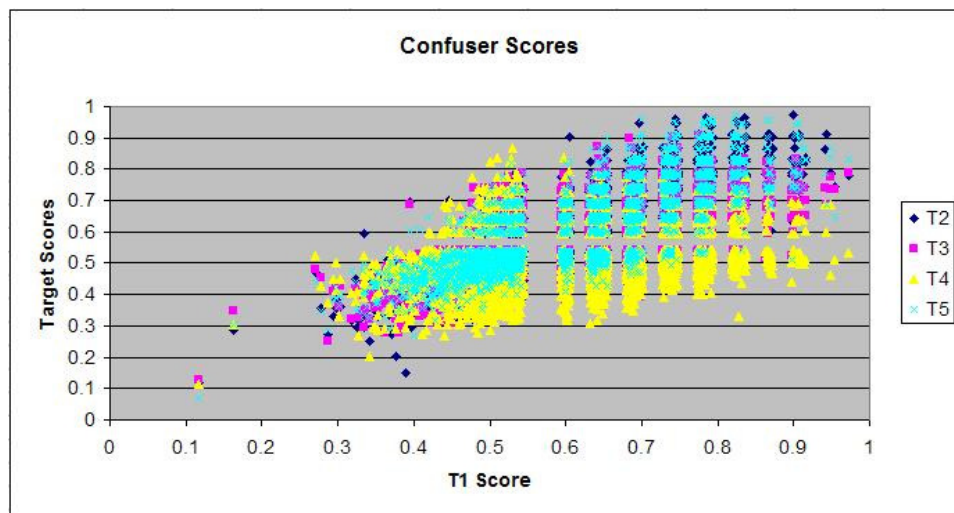


Figure 5. T1 (Confuser or Clutter) Target Likeness Scores.

The plots show how the true target can be confused with the other targets. For instance, Figure 4 shows how T2 (Tiger) is often given a higher “T1” (“Clutter”) score, but typically a lower “T4” (“T-72”) score. Thus, any Tiger encountered will likely be confused with Clutter, but never with a T-72, based on TLS. The graphs also show how the scores range from zero to one, but they do not add up to one. In some cases,

even invalid scores can be found. These indicate missing data or bad readings.

Invalid scores are dealt with later in this study.

The test file contained 3893 runs. Each run is against an actual target that was recognized. Actual targets include In-Library (IL) targets of T2, T3, T4 and T5 and Out-Of-Library (OOL) targets or confusers of Con1, Con2, ... Con11. T1 is not a true target since T1 does not exist in the real world. As stated above, T1 is used to compare found objects to a fuzz ball in the hopes of eliminating clutter. Since the confusers are OOL, and an identification of T1 does not result in a target registering on the view screen, any confuser found as T1 will be deemed a correct identification for the purposes of this study. Later studies are suggested to treat confusers individually.

Algorithms and Parameters

Five algorithms were suggested for this study. The output of the study should be understandable presentations of all meaningful information so the decision-maker can select which algorithm is the best. The algorithms are originally coded as A1, A2, A3, A4 and A5, but will be assigned the meaningful names of Biggest-TLS, TLS-Bound, TLS & D, Big-D, and D-Bound, respectively, for this study.

Each algorithm's decision can either be "Forced" or "Not Forced" (Sadowski, 2005). A Forced decision means the algorithm will return an item in the target library. A return of "T1" will not be displayed since "T1" indicates "Clutter." A decision that is Not Forced may return a value of "Unknown" if certain algorithm-based criteria are not met.

Algorithms use Target Likeness Scores, Deltas, and thresholds:

- A Target Likeness Score (TLS), as defined above, is a value between zero and one that refers to estimates of target probability (Parker, 2005).

- A minimum TLS-Threshold is a value determined for each In-Library target. For algorithms A2 and A3, if the highest TLS Score does not meet this value, then “Unknown” is returned. For other algorithms, the TLS-Threshold defines the Delta value.
- A Delta score is the difference between a score and the TLS-Threshold for each target.
- A minimum Delta-Threshold is another value determined for each target. For algorithm A5, the largest Delta value must exceed the next highest Delta by this value or an “Unknown” is returned.

The algorithm solutions depend on the TLS, the TLS for the other targets, the minimum TLS-Thresholds, and/or the minimum Delta-Thresholds. Since only the minimum thresholds are used, minimum will be implied when discussing these thresholds. These parameters allow for algorithms to be customized to specific situations or battlefield environments. For instance, in a high clutter, low target area, the threshold for “Clutter” might be lowered. Thus, any score for “T1” will most likely meet the threshold, and any Delta created (score minus TLS Threshold) will likely be higher than all other Deltas. If a target is unlikely, TLS Threshold can be increased to prevent low score declarations of the item. The way each algorithm uses these values is as follows (Sadowski, 2006):

- A1 or Biggest-TLS: (Forced) The target with the highest TLS is returned.
- A2 or TLS-Bound: (Not Forced) The target with the highest TLS is returned if it meets the TLS-Threshold. Otherwise, “Unknown” is returned.

- A3 or TLS & D: (Not Forced) The target with the highest TLS is returned if it has the largest Delta and meets the TLS-Threshold. Otherwise, “Unknown” is returned.
- A4 or Big D: (Forced) The target with the highest Delta score is returned.
- A5 or D-Bound: (Not Forced) The target with the highest Delta is returned if it meets the minimum Delta-Threshold. Otherwise, “Unknown” is returned.

The output for each algorithm must be computed within the UIT. For each run, and each algorithm, the tool must compute what is returned: an In-Library target or “Unknown.” Other algorithms should be added for comparison in future studies. Future algorithms may apply fusion techniques to either the scores or to the algorithm solutions.

Optional Parameters

The data scores and five algorithms are given, but some parameters start undefined. For instance, although the algorithms are given, the thresholds that determine their outcomes are not. Also, the populations of each target in the environment are not given. Finally, the costs for misidentifications are unknown and may be difficult to obtain or difficult to agree upon.

Thresholds are determined individually for each target, but they may not be decided independently of the other targets. One suggestion is that a low, medium, and high threshold for each target be allowed. Another is that Receiver Operating Characteristic (ROC) curves be created that span possible thresholds. Alternatively,

the end user could be allowed to dynamically adjust each threshold. Later in this thesis, the use of dynamically responsive output to adjusted thresholds will be shown.

Target populations, called priors, in real-world situations may not be known with certainty. For the purposes of this study, the user will be able to enter a minimum, maximum, and most likely (median) number for the population of the target to be encountered.

Costs would be helpful in determining the best algorithm. By identifying a cost with each misidentification, an expected cost can be created for each algorithm. However, determining costs are subjective and can vary widely from one user to the next. Thus, a cost matrix will not be the focus of this study, but research is included for using one, provided all targets can be designated as, “Friend,” “Target of the Day,” “Other Hostile,” or “Neutral” and costs can be obtained.

Output

The optimal output would be a list of algorithms from best to worst, if algorithms could dominate each other. If each algorithm has unique tradeoffs, a graphical depiction of these tradeoffs would be next best. However, tradeoffs are up to the decision-maker, so the output at this stage of the tool is a meaningful representation of the measures available. These include ROC curves, Classifier Evaluation Quantifiers and a Cost Matrix. In addition, two new quantifiers and presentation methods, Vertical ROC curves and Prior curves, are developed to answer questions the previous methods could not. These will be discussed in later chapters, while the more common methods will be briefly discussed next.

ROC Curves

A ROC curve is a curve commonly employed to compare the two types of errors. Typically, a threshold is varied, and the True Positive is plotted against the

False Positive (Albrecht, 2005:63; Hopley, 2006). True Positives and False Positives will be discussed in Section III, Methodology. ROC curves are applied to tests with only two outcomes: “TRUE” and “FALSE.”

ROC curves were a suggested output, but they must be modified to account for the unique nature of this study. First of all, some algorithms do not have thresholds that can be varied. Without a threshold, they would simply form one point on the ROC space. Also, more responses than “TRUE” and “FALSE” are possible and some misidentifications are more costly than others. Furthermore, with “TRUE” being different for each target being recognized, one ROC curve could be created from each individual target. Finally, solutions depend on more than one score and more than one threshold. One fixed score cannot represent reality due to this dependency, just as lowering only one threshold cannot guarantee a response changing from “FALSE” to “TRUE.” Since solutions depend on the values of every available threshold, thresholds to generate a ROC curve should only be varied one at a time. The dependency of the data will be shown in detail in Section III, Methodology.

Summary Data

In addition to ROC curves, summary data was suggested to provide classifier evaluation quantifiers. These quantifiers – Precision, Recall, F-Value, Accuracy and Mutual Information – will be examined in detail in Section II, Literature Review, and Section III, Methodology. An example of this information is given in Table 2. As with the ROC curve, these quantifiers must be modified for use in a multiple declaration problem. We propose that this information be computed for reference, although it should not be the primary quantifier due to how misleading some of these quantifiers can be.

Table 2. Classifier Evaluation Quantifiers.

	A1	A2	A3	A4	A5
Precision (Sum)	2.1	2.5	3.2	2.1	3.4
Recall (Sum)	3.3	3.5	2.8	2.4	2.3
F-Value (Sum)	1.7	2.3	2.6	1.8	2.3
Accuracy	0.23	0.36	0.51	0.16	0.53
Mutual Information	0.08	0.11	0.09	0.06	0.06

Cost Matrix

A summary cost was also suggested. Originally a Cost Matrix was developed based on the population data provided. This matrix is highly dependent on accurate estimates of population and on cost. Estimates of population can follow a triangular distribution with only the minimum, maximum, and most likely values being speculated. Also, costs can be purely subjective. Thus, ROC curves, classifier evaluation quantifiers – as well as alternatives such as Vertical ROC curves and Prior curves to be introduced in Section III, Methodology – will be the quantifiers for algorithm comparison in this study.

Research Objectives/Questions/Hypotheses

The objective of this research is to find a useful method for selecting the best ATR algorithm available given a set format of data or for presenting meaningful information for the decision-maker to make the selection. In addition, this research will present an option to determine better parameters for the ATR Algorithms.

Research Focus

The primary focus of this research is to exploit the capabilities of Microsoft Excel and Visual Basic to create a programmable method of generating visual information from summary statistics on a given set of data. Although most of the

information presented is computed from conditional probability, new methods have been employed to present all the data. In addition, the research touches upon the handling of “Unknown” responses, missing data, and multiple error calculations. Some new techniques, the Vertical ROC curve and Prior curve, are developed to better compare the differences between algorithms. It is suggested that linear programming and fusion techniques be explored in future research to increase the effectiveness of this tool.

Methodology

The UIT takes a set of data in a given format and processes it through MS Excel using VBA code. Further parameters on the five algorithms and the current operational environment are then asked for and entered by the user. Information is presented visually throughout a number of MS Excel spreadsheets. Quantifiers and graphs are generated on the spot. These include ROC curves, Vertical ROC curves, Prior curves and Classifier Evaluation Quantifiers. The parameters for the five algorithms and the current operational environment can then be changed to improve the algorithm responses, to answer sensitivity questions or to meet future needs of the user. Calculations and corresponding visual information on each sheet of the Excel file will automatically update with new user inputs.

Assumptions/Limitations

The UIT was built using a set of data provided by the customer and many assumptions are made from the sample data file. The UIT assumes that all following data will be provided in the same format. Some specifics in format include placement of the data in the data file and the location of In-Library targets in the heading. It is assumed that the In-Library targets are listed in priority from left to right. It is

assumed that a Clutter target designated “T1” will always be present. It is assumed that Confusers will be identified with a name starting with “Con.”

Confusers are objects that may be mistaken for In-Library targets. The ATR algorithms were tested against a large number of confusers. Unless otherwise stated, the correct identification for confusers is assumed in this walkthrough to be “Clutter” with “Unknown” also acceptable. In addition, a response of “Unknown” due to ties and non-declarations will be synonymous with “Clutter.” This makes a major difference in error calculations as “Unknown” responses are not only considered but are counted as misidentifications for all non-clutter targets.

Priority is given in the UIT to the leftmost targets in the data file’s heading. Thus, ties between targets are broken by returning the target highest in the priority list. For example, if the columns in the data file have “Clutter,” “Tiger,” and “Bradley,” listed from left to right, then ties involving “Clutter” returns “Clutter” while a tie between returning “Tiger” or “Bradley” returns “Tiger.” Although a response of “Non-Declaration” is arguably appropriate, assigning an actual target as a response shows the algorithm met a declaration requirement for at least one target.

A primary assumption of the UIT is that all data sets will include “Clutter” as an In-Library target and as the target with the highest priority. This allows “Forced” algorithms to have “Unknown” as a valid declaration. This is appropriate as a large number of the test samples are runs against confusers. Not allowing for “Clutter” would put “Forced” algorithms at a distinct disadvantage in clutter heavy environments which would make them impractical if this was not allowed.

Implications

The UIT allows users to choose between multiple selection criteria and adjust parameters as needed. Programmers are concerned with the probability a system works properly. They are concerned with Horizontal Measures of Performance (MOPs) or the probability of declarations given a state of the system. However, operators are concerned about Vertical MOPs, the probability of a state of the system given a declaration. Vertical MOPs can be poor even if the Horizontal MOPs are good. This can occur when the probability of the state of the system, or priors, is close to zero for the targets with the good Horizontal MOPs, and large for targets with poor Horizontal MOPs. Multiple selection criteria allow effectiveness to be analyzed from both perspectives. Although ROC curves can provide information on Horizontal MOPs, the criteria developed in this UIT allows for Vertical MOPs to also be provided in graphical form using a modified ROC curve referred to in this thesis as the Vertical ROC curve. Since Vertical MOPs are very sensitive to priors, Prior curves are also developed to answer what-if questions and perform sensitivity analysis. Vertical ROC curves and Prior curves provide a unique look that is quick and meaningful to the operator, but depend heavily on the operational environment.

Effectiveness of a system also varies with user-defined parameters. By allowing a dynamic tool for analysis, the effects of changes to the environment will be presented as soon as they are entered into the spreadsheet. Furthermore, sensitivity analysis is made possible through these adjustments and by allowing a range of prior probabilities to be entered.

Preview

Research began by investigating which methods of fusion, summary statistics, and information presentation could be used on the data provided. These methods are outlined with recommendations for further study in the Section II, Literature Review. Once methods were explored, the data provided was examined for possible complications and implications.

Both the data and possible outputs for the data are examined in the Problem Statement earlier in this section. Although problems such as missing data and tool limitations have been identified in the Problem Statement, the section on Methodology, Section III, provides information on each problem resolution and rationale as to why assumptions are made using a comprehensive walkthrough. Output calculation is discussed in depth in this section as well as the usefulness of each selection criteria. New tools, the Vertical ROC curve and Priors curve, are introduced and modifications to old tools, ROC curves and classifier evaluation quantifiers, are discussed as well in this section.

Section IV, Results and Analysis interprets the findings of the UIT in a sample analysis. Speculations are made and theories investigated throughout the section as if made by someone using the tool. The general effectiveness and relevance of the tool along with suggested future research is given in Section V, Discussion.

II. Literature Review

Historical Perspective

A User Interface Tool (UIT) is needed to allow easy comparison between Automatic Target Recognition (ATR) algorithms. Developing this tool required an examination of research on classifiers, error, and decision-making processes. Also researched was what mathematics is being applied to compare and select the best ATR algorithm. Methods for ATR algorithm comparison and selection and how they can be applied to the problem at hand were a primary interest in the literature review.

Error in Hypothesis Tests

Commonly, error is measured in hypothesis tests or tests to determine whether a statement, the null hypothesis, should be rejected or should fail to be rejected. The actual test is conducted on the alternate of the null hypothesis. Errors of two types are typically reported. The first error is called a False Positive, Type I error, or Alpha error, and it is the probability of rejecting the null hypothesis when the statement is true (Hopley, 2006). The second error is called a False Negative, Type II error, or Beta error and is the probability of not rejecting the null hypothesis when the alternate hypothesis is true (Hopley, 2006).

For clarity, some definitions shall be tailored to make them more applicable to the objective of this thesis. These definitions shall be defined within each chapter.

The null hypothesis in ATR is the statement a detected object is, in actuality, a specific In-Library target. Not rejecting the null hypothesis implies the algorithm identified the object as the In-Library target, so the symbol for that target will be displayed for the end user. Not rejecting the null hypothesis implies accepting it as truth as far as the end user, the pilot, is concerned. Thus, rather than using the phrase,

“not rejecting,” the null hypothesis will either be accepted or rejected in this paper. A special case in which “NON-DECLARATION” is also allowed as an alternative to accepting or rejecting the hypothesis will be discussed further in the literature review under the topic, “Error with ‘Unknown’ Allowed.”

As the null hypothesis will either be accepted or rejected, the case where it is accepted will be referred to as returning the statement, “TRUE.” When it is rejected, it will be referred to as returning the statement, “FALSE.” If the null hypothesis is true, this analysis will refer to the state of the system as TRUE without quotes. If the statement is false, the state of the system is FALSE. If the test is more specific – for instance if the test is to determine if the object is a HOSTILE, NEUTRAL, or FRIEND – the returning statement will be the classification label in quotes.

With these new definitions, an Alpha error is a declaration of “FALSE” when the system is TRUE. A Beta error is a declaration of “TRUE” when the system is FALSE. When the analysis becomes even more specific, the notional names of the individual targets will be used instead of TRUE, FALSE, “TRUE,” and “FALSE.”

Receiver Operating Characteristic

A visual method typically used to display error is the Receiver Operating Characteristic (ROC) curve (Albrecht, 2005:63). A ROC curve plots a test’s False Positive value versus its True Positive, or the probability of returning “TRUE” when it is TRUE, value (Albrecht, 2005:63). The ROC curve creates the plot by adjusting the threshold in determining whether to reject it is “TRUE” or not. It plots both values at multiple thresholds. ROC curves are useful in comparing many algorithms by showing visually which one outperforms the other with better True Positive versus False Positive rates. Figure 6 displays a ROC curve that compares three algorithms:

Algorithm 1, Algorithm 2, and Algorithm 3. The further up and left the curve bends, the better the algorithm (Simon, 2005). In this example, Algorithm 1 and Algorithm 2 are almost identical, and both outperform Algorithm 3. Algorithm 3 is only preferred over the baseline where False Positives and True Positives have an equal tradeoff.

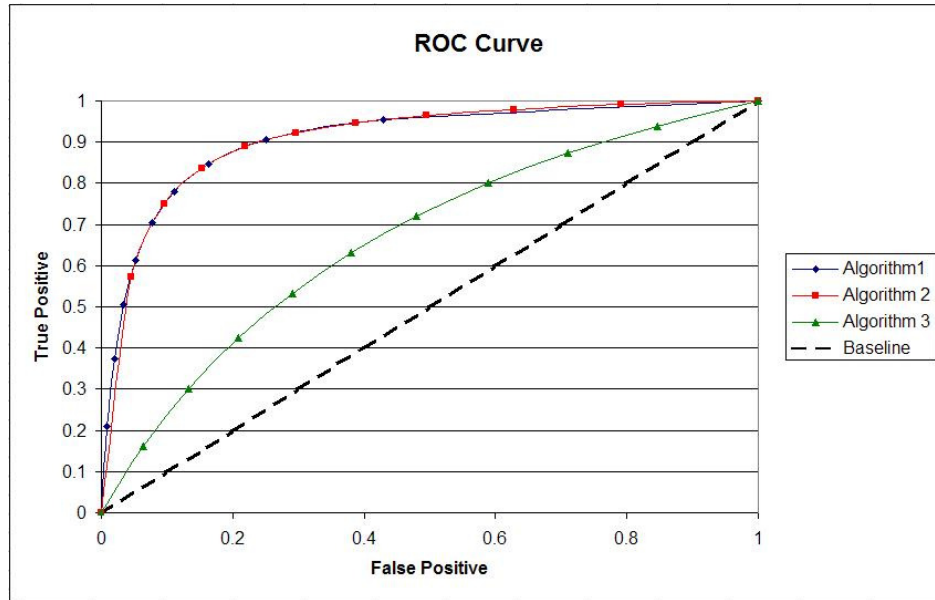


Figure 6. Receiver Operating Characteristic Curve comparing three algorithms.

Error in ATR Comparisons

Historical measures to compare the difference between ATR algorithms are called Horizontal and Vertical Measures of Performance (MOPs) (Sadowski, 2006).

Horizontal MOPs are probabilities of declarations given a state of the system. Horizontal MOPs are more of a concern to engineers and programmers as to whether a decision algorithm is working properly and displaying the correct result. Typical Horizontal MOPs include the probability:

- True Positive: a system outputs “TRUE” when its input is TRUE
- True Negative: a system outputs “FALSE” when its input is FALSE
- False Negative: a system outputs “FALSE” when its input is TRUE

- False Negative: a system outputs “TRUE” when its input is FALSE (Hopley, 2006)

Vertical MOPs are probabilities of the state of the system given a declaration.

Vertical MOPs are more a concern to the user who sees the end result of an algorithm.

These MOPs measure whether or not the result is accurate. The probabilities in

Vertical MOPs rely on the total population of TRUE and FALSE system states.

Typical Vertical MOPs include the probability:

- Positive Predictive Value (PPV): the state of the system is indeed TRUE when the display reads “TRUE” (Wikipedia, 2006)
- Negative Predictive Value (NPV): the state of the system is indeed FALSE when the display reads “FALSE” (Wikipedia, 2006)
- the state of the system is actually FALSE when the display reads “TRUE”
- the state of the system is actually TRUE when the display reads “FALSE”

The difference between Horizontal and Vertical MOPs is readily found in the ATR environment. For instance, consider the situation where a hostile object is detected. An ATR algorithm will test whether or not the object is “HOSTILE.” In this case, a return of “TRUE” indicates “HOSTILE” and a return of “FALSE” indicates “FRIEND.” A programmer might prefer an algorithm that correctly identifies targets, both friends and foes, eighty percent of the time. However, in an environment where more HOSTILES are encountered, an algorithm that is more likely to make mistakes in favor of classifying any target as “HOSTILE” would be more accurate to the user. This accuracy is because less of the abundant HOSTILES will be classified as “FRIEND.” A higher percentage of any FRIEND it encounters is misclassified as “HOSTILE,” but less FRIENDs are encountered.

By changing thresholds due to the probability of encountering a specific target (also known as a prior probability or priori) (Albrecht, 2005:174) the Vertical MOPs can improve, but the Horizontal MOPs will also change. However, prior probabilities are difficult to obtain. Also, dangers can arise from changing an algorithm to meet one measure of performance. These dangers and further comparisons of Horizontal and Vertical MOPs are explored further in the literature review under the topic “Possible Algorithm Performance Measures.”

Feature Determination

The classification process being researched in this study involves reporting a decision from features. A feature is a measurement extracted from a set of data (Roli, 2002:10). In particular, the feature scored in this project is the “likeness” a detected object is to any given target in the library. The “library” refers to the set of targets on which the system has information. The “likeness” score is not the probability an object is the target, but how “like” it is to the target. More specifically, it “refers to estimates of target probability that are output from a target detection system under test (SUT)” (Parker, 2006). A detected object has a “likeness” score, referred to in this study as Target Likeness Score (TLS), computed for each target in the library, so each run has a set of features calculated.

In the data provided, a library of five targets was given. It was not designated whether each target was friend or foe. In fact, the first target, T1, is actually a general “clutter” target (Sadowski, 2006) which is used to compare the detected object to a non-target, like trees and bushes. The other targets were designated T2, T3, T4, and T5. As stated in Section I, Introduction, for simplicity the names T1, T2, T3, T4 and T5 will

be replaced with the more meaningful names of Clutter, Tiger, Panzer, T-72 and Bradley respectively.

Features were computed during runs where an intended object was detected. Each run had the following information recorded: the actual object, details about the run and a likeness score to each of the targets in the library. An example of two of these runs and the computed scores for each target is provided below in Table 3.

Table 3. Four runs from the sample data file repeated from Table 1.

	A	B	C	D	E	F	G
1	File_Name	Actual	T1	T2	T3	T4	T5
2	5_D1_1536_2048	Con1	0.513	0.517	0.525	0.393	0.601
3	5_D1_1536_2048	T2	0.48	0.596	0.421	0.372	0.457
4	5_D1_1536_2048	Con2	0.691	0.467	0.528	0.371	0.512
5	5_D1_1536_2048	Con3	0.632	0.651	0.519	0.502	0.53

Decision Determination

Current algorithms make a decision based on the given features. The decision is restricted to classifying the object as one of the targets in the library (IL), classifying the object as something out of the library (OOL), or classifying it as a non-declaration. A non-declaration indicates the target is a target and not clutter, but it cannot distinguish which target it is.

Five algorithms are being considered to make a decision based on the features. Each algorithm was referred to as A1 through A5, but given more meaningful names of Biggest-TLS, TLS-Bound, TLS & D, Big D, and D-Bound, respectively. The algorithms in Figure 6 do not correspond with these algorithms. The algorithms are defined as (Sadowski, 2006):

- Biggest-TLS (A1): Return the In-Library target with the highest TLS for that run.

- TLS-Bound (A2): If a run's biggest TLS is greater than a specified amount called the minimum TLS-threshold for that target, then return that target. Otherwise return "Unknown."
- TLS & D (A3): Compute the Delta score for each target. The Delta is equal to the TLS for that target minus the specified minimum TLS-threshold for the target. If the Delta is negative, return "Unknown". If the highest TLS is greater than the minimum TLS-threshold and the Delta for that feature is greater than the Delta for all other features, return that target. Otherwise return "Unknown."
- Big D (A4): Compute the Delta score for each target. Return the In-Library target with the highest Delta score for that run.
- D-Bound (A5): Compute the Delta score for each feature. If the highest Delta score is greater than the Delta score for all other features by a specified minimum Delta-threshold, return that target. Otherwise return "Unknown."

Biggest-TLS (A1) and Big-D (A4) are "Forced" to return an In-Library target once it detects an object. The other algorithms are allowed to return "Unknown" so they are "not forced" (Sadowski, 2006). In environments with excessive amount of clutter that will be detected and considered for classification but will be found not in the library (NIL), the "forced" algorithms are more likely to perform poorly, since they must classify any detected object as one of the targets. To help rectify this situation, one of the targets, T1, is a clutter target (Sadowski, 2006). T1 does not represent a real target. Instead it represents common clutter in the area. If an object is classified as "T1," no symbol will be displayed as the object will be identified as Clutter (Sadowski, 2006). Including T1 reduces the error of "Forced" algorithms, but it cannot reduce all of the error since different clutter objects may not have similar features.

User-Defined Input Variables

More information is required before a decision can be made and error measured. The first missing information is the threshold levels. “Likeness” threshold levels, or TLS-thresholds, are required before both TLS-Bound (A2) and TLS & D (A3) can compute a decision. The TLS-threshold is also used to calculate Delta scores, as a Delta score is the difference between a target’s TLS and TLS-threshold. A Delta-threshold is required for D-Bound (A5). These thresholds are similar to those used in the ROC curve, as decreasing the threshold allows for more True Positives but results in more False Positives.

One approach would be to vary thresholds in levels – low, medium, and high – depending on the likelihood of having clutter in any given image (Sadowski, 2006). Another would be to vary thresholds based on costs for misidentification. This method is further examined later in the literature review under the topic “Fusion Comparisons.” Thresholds can also be based on historical data or a training sample of test runs.

Other missing information includes estimated populations of the targets, or an estimated distribution of the population of the targets, and costs for misclassifications. This information is missing because in a region with military actions, these values can constantly change. The sponsor requested an ATR UIT which allows the user to define these values. For the UIT, the user-defined variables shall be:

- Operating points or levels of thresholds
- Scenarios or levels of clutter
- Prior populations or probability of encountering each target.

Follow-on projects may expand the scope of user-defined parameters. For example, in the future, a user may want to define alternate algorithms using more parameters for comparison. Other user-defined parameters could include weights, or costs, of correct and incorrect classifications.

Application of Historical Perspective

Most of the historical techniques can be applied with some alterations to the ATR UIT project at hand. With five targets rather than TRUE and FALSE, the error tests do not seem applicable. With more than two classifiers, ROC curves can also be difficult or impossible. However, if the targets are converted from the labels “T1,” “T2,” “T3,” “T4,” and “T5” to an alternate label of “HOSTILE” or “FRIEND,” the error tests and ROC curves can be applied. In addition, it will be shown that adding “Unknown” as a class still allows ROC curves to be used (Albrecht, 2005).

Alternatively, each target’s ROC curve can be plotted individually. To measure both Vertical and Horizontal MOPs, a modified ROC curve called a Vertical ROC curve shall be applied in this project. A Vertical ROC curve is developed by applying some principles of the ROC curve to Vertical MOPs rather than Horizontal MOPs.

The current way of computing features and decisions are useful but may be improved upon. Features currently provide one-look per test run. Follow-on research may be accomplished to allow multiple sensor looks and use sensor, data, and time-series fusion to compute features. Fusion might provide more distinct features scores by using information from the other features to compute the score. Decision computation can be improved through the use of feature fusion and decision fusion. For this study, however, analysis is limited to only the data provided.

Possible Algorithm Performance Measures

Another concern in selecting the best algorithm is how to prove it is the best algorithm. To compare algorithms, an appropriate evaluation quantifier should be chosen.

According to a study conducted by Hanna M. Wallach at the University of Cambridge, classifiers can be compared using Accuracy, Precision, Recall, F-score and Mutual Information (Wallach, 2004). Accuracy can be defined as the combined probability of True Positives and True Negatives. Precision is another name for Vertical MOPs. Recall is another name for Horizontal MOPs. An F-Score is a “harmonic mean” between the two whereas mutual information, “is defined as the Kullback-Leibler divergence between the joint distribution of <output> y and <truth> t and the product of their marginals:” (Wallach, 2004)

$$I(y:t) = D(P(y,t) \parallel P(y)P(t)) = \sum_y \sum_t P(y,t) \log \frac{P(y,t)}{P(y)P(t)} \quad (1)$$

Wallach compares three models to determine which quantifier is the best. Based on her study, Accuracy, Precision, Recall and F-Score can be misleading. An example of this is when ninety percent of the objects in a Region of Interest (ROI) are the target T2. Then, any algorithm that simply labels all detected objects as “T2” will have ninety percent accuracy, ninety percent precision, high Recall and high F-Score. However, the algorithm is completely dependent upon the population of T2, so it provides little useful information. Rather than these classifier quantifiers, Wallach suggests using Mutual Information and then the other classifier quantifiers.

Alterations are necessary to apply this research to a scenario with six classes: “T1,” “T2,” “T3,” “T4,” “T5,” and “U.” Precision and Recall could be calculated for each individual class. From these values, F-Score can be calculated for each class. A report can show individual, average or the sum of these values. The sum of these values for the algorithms and test data are shown in Table 4.

Table 4. Classifier Evaluation Quantifiers.

	A1	A2	A3	A4	A5
Precision (Sum)	2.1	2.5	3.2	2.1	3.4
Recall (Sum)	3.3	3.5	2.8	2.4	2.3
F-Value (Sum)	1.7	2.3	2.6	1.8	2.3
Accuracy	0.23	0.36	0.51	0.16	0.53
Mutual Information	0.08	0.11	0.09	0.06	0.06

Error Exploration

Measuring error also changes with multiple class labels. Error with two class labels and an “Unknown” class label can be considered as a three-class system. Error for an unlimited number of targets can be refined into a smaller system by reclassifying each target as one of a limited number of labels. Alternatively, the error for each target can be considered separately. Each of these methods for dealing with error is considered in this study.

Error with “Unknown” Allowed

Allowing an “Unknown” label can be both helpful and harmful. “Unknown” can be defined as two different situations. One situation is where the detected object is not like any target in the library. This is the case where the detected object is most likely an Out-of-Library (OOL) confuser or Clutter. However, “Unknown” can also occur if a detected object is equally like two targets in a library. This case can be

referred to as a “NON-DECLARATION,” where the detected object is found to be a target in the library, but the algorithm is uncertain which one. Possible ways to deal with a “NON-DECLARATION” is to display a symbol that toggles between the two In-Library targets or to display “Unknown.” In the case where, “Unknown” is decided to be an OOL confuser, the system can display no symbol.

In the case where detected objects are always in the library, “Unknown” can significantly change the determination of True Positives and True Negatives. “Unknown” signifies a level of confidence is needed to make a final decision. On one hand, it is not incorrect to call an object “Unknown.” Thus, if these classifications are taken out of the calculation, values for False Positives and False Negatives decrease. However, both values for error would be zero if all objects could be declared as “Unknown,” but no correct identifications would be made. Since “Unknown” is also not correct, a penalty could be assessed to these declarations to limit its use. Finally, since declaring an object as “Unknown” might be like declaring the target as “Clutter,” it is reasonable to treat “Unknown” as just another target. Each of these methods of resolving “Unknown” have their own pros and cons.

Alternatively, a three-dimensional ROC curve could be constructed. A two-dimensional ROC curve would plot True Positives versus False Positives. A three-dimensional ROC curve could add a third axis showing percent of declaration (Albrecht, 2005). Such a plot would show how the ROC curve changes as the confidence needed to make a declaration increases. An example of such a plot is shown in Figure 7.

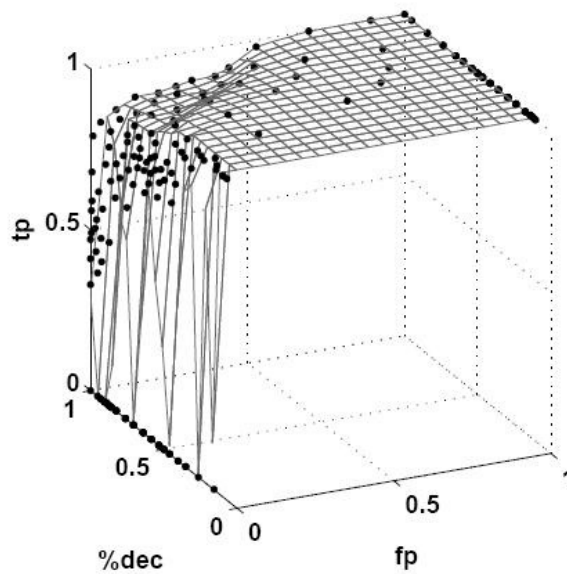


Figure 7. Surface plot of ROC Curve. (Albrecht, 2005:98)

The surface plot shows changes over different percent declarations. The percent of False Positives decrease as declarations are not forced. Alternatively, the percent of True Positives increase until the percent of declarations is zero. This plot is useful in showing the impact of increasing thresholds to the decision between two classifications.

Error with Multiple Classifications

Although three-dimensional ROC curves are useful if only two classifications other than “Unknown” are possible, the sample of interest contains six classifications from five algorithms. Future data could have a larger number of targets in the library. Providing a surface plot for each algorithm and each target would not provide a straightforward, visual selection process for the user. A method of handling multiple classes needed to be explored.

Suggested from a study conducted by Timothy W. Albrecht, Major, USAF at the Air Force Institute of Technology, one method could involve categorizing each target as one of a limited number of labels (Albrecht, 2005). Rather than comparing each individual target, an appropriate label indicating the meaning of the target can be used. Thus, rather than a list of targets, targets could be labeled under the appropriate category:

- Target of Day (TOD) : the hostile target of interest (Albrecht, 205:98)
- Other Hostile (OH) : targets that are hostile but not of interest
- Friend (F) : an allied target
- Neutral (N) : a target that is neither friend nor hostile
- Out-of-Library (OOL) : An unidentified confuser, or clutter
- Non-Declaration (Non-Dec) : An ambiguous target
- Unknown (U) : Either OOL or Non-Dec

Using categories fixes a maximum number of labels to be studied regardless of library size, possibly reducing the number of cost calculations. Cost calculations result from each correct and incorrect label having separate costs, and different incorrect labels can vary widely in costs. For example, if the target detected is a T3, then “T2” and “T4” are incorrect labels. If T2 and T3 represent friends while T4 is a hostile, declaring T3 as “T2” declares a friend incorrectly as a different “Friend,” while declaring T3 as “T4” declares a friend incorrectly as “Hostile.” Both results are incorrect, but the second cost could be much higher. Rather than setting a cost for all target combinations, placing targets in categories can reduce the calculations in the above example to the cost of misidentifying a friendly target as any friendly target versus identifying a friendly target as any hostile target. In such a way, cost

calculations can be reduced to five outcomes listed below and represented in a confusion matrix in Table 5. (Albrecht, 2005)

- Correct Label : Making the correct classification
- Critical Error : Classifying F or N as “TOD” or “OH” and classifying TOD or OH as “F” or “N” (Albrecht, 205:98)
- Non-Critical Error : Classifying TOD as “OH” and classifying OH as “TOD” and classifying U as “OH” or “TOD” (Albrecht, 205:97)
- Lessor Error : Classifying F or N as “U” and classifying U as “F” or “N” (Albrecht, 205:98)
- Non-Declaration : Classifying any target as “NON-DEC” (Albrecht, 205:97)

Table 5. Cost Confusion Matrix.

Cost Confusion Matrix						
	"Friend"	"Neutral"	"TOD"	"OH"	"Unknown"	"Non-Dec"
Friend	C	LE	CE	CE	LE	ND
Neutral	LE	C	CE	CE	LE	ND
TOD	CE	CE	C	NCE	NCE	ND
OH	CE	CE	NCE	C	NCE	ND
OOL	LE	LE	NCE	NCE	C	ND
C	Correct					
ND	Non-Declaration Error					
LE	Lessor Error					
NCE	Non-Critical Error					
CE	Critical Error					

Although OOL misidentification costs might depend on the OOL target, this set of outcomes provides an alternative for reporting cost. A cost could be assigned to each outcome and calculated or the probability of each of these outcomes could be reported. Thus, categorizing targets into these labels can allow for a limit to the number of truths and declarations to be analyzed regardless of library size and provides an easier way to calculate costs to compare and ultimately select algorithms.

Although useful for systems dealing with many targets, the sample data had only five targets. Costs and categorization would require additional user input. This study shall leave the question of costs up to future research.

Decision-Making Process

To select the best algorithm, the five suggested algorithms can be compared or a new algorithm can be offered. A new algorithm could use the raw data or any of the five algorithms' decisions as information to make its decision. To determine the best way to do make this decision, fusion and sampling was researched.

Multiple Classifier Systems

According to a tutorial created by Professor Fabio Roli at the University of Cagliari, the decision-making process can be broken into fusion steps: sensor fusion, data fusion, feature fusion, and decision fusion (Roli, 2002:28). Roli's work begins with sensor fusion and selecting the most appropriate classifiers using sensing, segmentation, and feature extraction. Unfortunately, the data provided for use in this thesis is a one-look set of scores already calculated from each run, so sensor fusion and data fusion is not in the scope of this project, but could be explored in follow-on research. However, classification and post processing, accounting for costs of errors, can be useful in deriving a better algorithm.

According to one expert, Hidden Markov Models (HMM) is another pre-feature fusion model that can improve Combat Target Identification (Albrecht, 2005:152). This system uses a time-series classifier design method to compare observations to a hidden state space before making a decision or a decision vector of confidence levels. This method could be highly useful in determining the correct classification. Unfortunately, using this method is outside the scope of this project since it requires

multiple sensor observations and is conducted before feature scores are determined.

Roli's work also involves selecting the appropriate decision boundary based on discriminant scores. Applying decision boundary calculations to training data could be useful in computing the most effective thresholds for an algorithm.

Decision fusion is typically accomplished through ensemble Multiple Classifier Systems (MCS), but it can also include modular MCS (Roli, 2002:48). Ensemble MCS compares the decision, task solution, alternative algorithms make. Modular MCS when incorporated with ensemble MCS instead compares the entire decision vectors – not only an algorithm's first choice, but the second choice, third choice, ..., last choice as well – from each algorithm to compute the best decision. Using this method, classifications could be eliminated serially or conditionally from consideration (Roli, 2002:29). Then, a voting technique could be used on the remaining classifications, as in ensemble MCS, or ranks and confidences assigned to the list of possible classifications. The hybrid ensemble/modular MCS method could be applied to the provided algorithms to make a better decision.

According to one expert, the use of ensemble MCS is dependent on complementary classifiers and error diversity (Roli, 2002). Complementary classifier means each classifier makes different classification errors. The provided algorithms being used are similar in how either the "likeness" score or the "Delta" score is used, so complementary classification is not valid at the decision level. Some of the algorithms are so similar that they simply add "Unknown" to the decision vector of a previous algorithm. Still the concept behind complementary classifiers may still be applied to the "likeness" and "Delta" scores the algorithms use.

With the restrictions on classifiers, methods for classifier ensemble design are restricted, but some of the principles can still be used for creating training sets. Training sets can be used to develop thresholds for the algorithms. Data-splitting, bootstrapping, and bagging are three common methods for training data manipulation (Roli, 2002:72). Data-splitting divides training sets into disjoint subsets. Bootstrapping allows random samples to be drawn from the training set with replacement. Bagging uses bootstrapping to create training sets of equal size. Training sets can create weights on features. A technique called Ada Boost describes creating weights for different training sets, then updating them based on misclassified patterns (Roli, 2002:70). Noise injection can be used when the sample size is smaller than the feature space, but with five likeness features, this is not an issue for this project.

Fusion Comparisons

A study was conducted at the Air Force Institute of Technology to compare techniques for decision level fusion, feature level fusion, and an intermediate level of fusion (Leap, et al.)

The decision level fusion, Identification System Operating Characteristic (ISOC), could be applied to the ATR project. ISOC can be applied to multiple classifiers and multiple output states. It uses conditional probabilities, rule selection, and costs to determine the best decision rule.

Another decision level fusion is the ROC Fusion (Leap, et al.). It begins with a training set and uses a test set to generate ROC curves of each classifier. From these ROC curves, an optimal threshold is determined. Based on these thresholds, a third set is tested to produce a final ROC curve. This method, however, may be more appropriate for systems with fewer output states.

An intermediate level fusion is given by a Probabilistic Neural Network (PNN) (Leap, et al.). PNN uses posterior probabilities as features. The feature level fusion is given by One Big Network (OBN). OBN fuses features in a Generalized Regression Neural Network (GRNN).

The study conducted by four experts showed that although fusion might not yield better accuracy, it can reduce miscalculations due to poor classifiers (Leap, et al.). In their study, the feature level fusion outperformed the other three types of fusion. Thus, GRNN may be a likely candidate for a good fusion model alternative to the provided algorithms. It must be kept in mind, as one expert noted, that the choice of models is not made on effectiveness alone but also on model complexity (Albrecht, 2005). The more complex the model, the more inefficient the algorithm may be for the system to provide timely information to the user.

III. Methodology

Input

The customer provided a list of scores in an MS Excel spreadsheet, suggested five possible algorithms, and asked for a way to compare all algorithms visually to determine which is best. The sample data file, entitled “AFIT_ATR_result_file.xls,” contained information on 3893 experimentation runs and a sample of it is shown in Table 6. This table shows the same sample from previous sections.

Table 6. Four runs from the sample data.

	A	B	C	D	E	F	G
1	File_Name	Actual	T1	T2	T3	T4	T5
2	5_D1_1536_2048	Con1	0.513	0.517	0.525	0.393	0.601
3	5_D1_1536_2048	T2	0.48	0.596	0.421	0.372	0.457
4	5_D1_1536_2048	Con2	0.691	0.467	0.528	0.371	0.512
5	5_D1_1536_2048	Con3	0.632	0.651	0.519	0.502	0.53

Each run has seven columns of information. The first is “File_Name,” which provides information about the run itself. The second is “Actual,” which shows what target was actually being examined during the run. The final columns are scores of either “#NAME?” or a value that ranges between 0 and 1. The value measures how like the actual target is to each of five targets – T1, T2, T3, T4, and T5 – currently in the library. The other response of “#NAME?” is returned when a system error in evaluating the actual target occurs. Samples that include the “#NAME?” response are given in Table 7.

Table 7. Sample Scores (Features) with invalid entries.

	A	B	C	D	E	F	G
1	File_Name	Actual	T1	T2	T3	T4	T5
2	5_D1_1536_2048	Con1	0.513	0.517	0.525	0.393	0.601
3	5_D1_1536_2048	T2	0.48	0.596	0.421	0.372	0.457
4	5_D1_1536_2048	Con2	0.691	0.467	0.528	0.371	0.512
5	5_D1_1536_2048	Con3	0.632	0.651	0.519	0.502	0.53
1129	12_D1_1536_2048	T5	0.695	0.696	0.69	0.508	0.793
1130	12_E1_1536_2048	Con1	0.74	0.527	0.533	0.451	#NAME?
1131	12_E1_1536_2048	T2	0.502	0.518	0.47	0.373	#NAME?
1132	12_E1_1536_2048	Con2	0.458	0.49	0.431	0.306	#NAME?
1133	12_E1_1536_2048	Con3	0.523	0.643	0.442	0.366	#NAME?
1134	12_E1_1536_2048	Con4	0.45	0.401	0.309	0.321	#NAME?
1135	12_E1_1536_2048	Con5	0.397	0.455	0.379	0.33	#NAME?
1136	12_E1_1536_2048	Con6	0.389	0.442	0.414	0.34	#NAME?
1137	12_E1_1536_2048	Con7	0.639	0.635	0.477	0.448	#NAME?
1138	12_E1_1536_2048	Con8	0.739	0.742	0.775	0.527	#NAME?
1139	12_E1_1536_2048	Con9	0.783	0.837	0.606	0.69	#NAME?
1140	12_E1_1536_2048	Con10	0.69	0.601	0.542	0.606	0.538
1141	12_E1_1536_2048	T3	0.634	0.631	0.738	0.517	0.645
1142	12_E1_1536_2048	T4	0.395	0.534	0.487	0.782	0.43

The responses in the “Actual” category include four In-Library targets – T1, T2, T3, and T4 – as well as eleven confusers, Con1 through Con11. T1 is missing from the list of possible Actual targets, because T1 exists only as Clutter. That is, T1 is an In-Library list of properties with general target characteristics created for the sole purpose of allowing a target to be classified as “Clutter” (Sadowski, 2005). T1 can resemble a tree or a bush or a more general fuzz ball. Thus, if any target is declared as a “T1,” then the target is declared as “Clutter” and is not displayed for the user. This allows algorithms with a forced declaration to still declare a target as “Out-Of-Library (OOL).”

The eleven confusers are synonymous with Clutter. They can be other vehicles that are likely to be encountered and considered targets by the identification

system. Unless otherwise stated, the correct identification for confusers is assumed in this walkthrough to be “Clutter” with “Unknown” also acceptable.

For clarification purposes, meaningful names are added to the five targets for the purposes of this walkthrough. T1 will be identified as, “Clutter.” T2 will be considered the Target of Day (TOD) and identified as, “Tiger.” T3 will be considered an Other Hostile (OH) target and identified as, “Panzer.” T4 will be considered a Friendly (F) and identified as, “T-72.” T5 will be considered a Friendly (F) and identified as, “Bradley.”

For simplicity, the scores will be referred to as Target Likeness Scores (TLS) as discussed in the previous sections. This shall prevent confusion with Discriminant Scores, since the tool shall not have knowledge as to how these scores, as well as future scores, are generated. This shall also prevent confusing these scores with Likelihoods.

Input Sheet Solutions

Along with the problem of invalid responses, the data file provided has other possible problems that must be resolved.

First of all, although it is easy to find any invalid responses visually, it is difficult for a macro to identify invalid responses. To identify invalid data, Excel displays “#NAME?” in the cell. However, this value is neither a number nor text as it is invalid. Thus numerical and text search criteria are unreliable. Instead, it is suggested that these entries be found manually by visually inspecting the data before the analysis. By manually examining the data for invalid entries, the reason why the data is invalid in the first place can be explored, allowing a better determination of how invalid data should be handled.

Different methods, each with their pros and cons, exist for resolving missing data problems. Two suggestions are either that a value is entered or the row be deleted. Deleting the row will allow for an analysis on only valid runs, but it may ignore a serious defect in the identification system that causes invalid returns and missing data. It can also hurt smaller data sets by deleting possible information. Alternatively, a substitute value may be entered in place of the invalid value. Possible substitute values are -1, 0, or the average response for the system from the data. Entering a negative one ensures no algorithm will choose the corresponding declaration as a response, but it will cause difficulties in creating scores later in the analysis. Entering a zero may still allow the forced algorithm using Delta scores to return the corresponding response if all other values are significantly below their minimum TLS-threshold. Allowing the average response to be entered allows the corresponding target declaration to still be considered, but it also makes the value of this run dependent on all other runs in the data set. This could be problematic in small-sized data sets. In our sample data with over 3000 runs, the average response will be used.

With TLS scores precise only to the third decimal point, ties can occur and do occur. Some algorithms base their response on the largest TLS score. If a tie occurs, there is no defined way to break the tie. Possible returns include “Unknown” or one of the In-Library targets. As stated earlier, “Unknown” can be interpreted as “Out-Of-Library” or “Non-Declaration.” In this case, “Out-Of-Library” may not be appropriate if the TLS scores are high or very appropriate if TLS scores are low. “Non-Declaration” is more appropriate, because the algorithm cannot choose between two valid returns. However, returning “Unknown” confuses whether it is due to a tie or due to low likeness scores. Instead, the User Interface Tool shall break the tie by using

the order targets are listed in the library. The earlier a target is in the order, the higher priority it will receive. In our sample data file, the targets are listed: “Clutter,” “Tiger,” “Panzer,” “T-72,” and “Bradley.” Thus, a tie between “Clutter” and “Panzer” will be resolved as “Clutter.”

Finally, although the list of In-Library targets is readily apparent and available, the list of Out-Of-Library targets is not. A list of Out-Of-Library, or Actual in the case of a data file, can be compiled manually, however in a real test these could be unknown. Currently, the UIT handles the list of Actual and the list of In-Library targets as follows:

- All targets identified as confusers in the list of Actual targets will begin with the string “Con.”
- “Clutter” will be an appropriate response for a confuser target
- “Clutter” will be the first item in the In-Library target list

Identifying all confusers with “Con” should not cause any problems. Since “Clutter” is equivalent to “Unknown” and confusers are Out-Of-Library, classifying a “Clutter” response for each confuser is appropriate. The third condition, however, assumes all target identification systems will have “Clutter” in their libraries and will give this response priority. This may not be true in all cases, so a better solution for this problem should be found in later updates to the UIT.

Master Worksheet

The User Interface Tool (UIT) has size limits, so a tool that identifies these limits and spawns other spreadsheets for analysis is most likely necessary. One way to accomplish this is to allow for a spreadsheet that can read in data and spawn another

spreadsheet with the requested information. The current master spreadsheet has a user interface button that allows for a spreadsheet to be opened from the tool. The interface is shown in Figure 8.

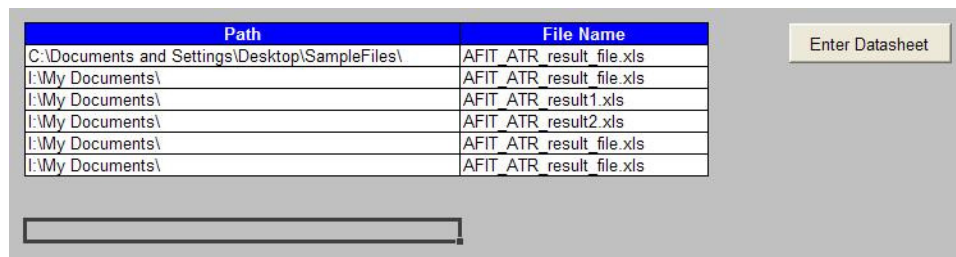


Figure 8. Interface panel for opening a spreadsheet to be analyzed.

Before the analysis can be started, the user must enter more information about the data. Information regarding the five suggested algorithms must be obtained. Further information on the populations of the possible targets must also be entered. Additional information on the orientation of the targets – for instance, whether a specific target is hostile, friendly, or neutral – would help but is not necessary at this stage of the tool. Costs may also be wanted, but shall not be used at this time. To enter these parameters, an understanding must be reached on the five algorithms suggested. These algorithms shall be covered in the following section.

User-Defined Parameters

Five algorithms were suggested by the customer. These algorithms were defined earlier as follows:

- A1: (Forced) The target with the highest TLS is returned
- A2: (Not Forced) The target with the highest TLS is returned if it meets the minimum TLS-threshold. Otherwise, “Unknown” is returned.

- A3: (Not Forced) The target with the highest TLS is returned if it has the largest Delta and meets the minimum TLS- threshold. Otherwise, “Unknown” is returned.
- A4: (Forced) The target with the highest Delta score is returned
- A5: (Not Forced) The target with the highest Delta is returned if it meets the minimum Delta threshold. Otherwise, “Unknown” is returned.

As stated in previous sections, for clarity each algorithm will be given a more meaningful name. The return for each algorithm is still which In-Library target – Clutter, Panzer, Tiger, T-72 or Bradley – the algorithm chooses. The Unforced algorithms are allowed to return an “Unknown” whereas the Forced algorithms use “Clutter” to designate unknown. The names for each algorithm as well as how the return is generated are rewritten below.

- A1: “Biggest-TLS.” Highest TLS.
- A2: “TLS-Bound.” Highest TLS, if it meets the TLS threshold.
- A3: “TLS & D.” Highest TLS, if it has the highest Delta and meets the TLS-threshold.
- A4: “Big D.” Highest Delta.
- A5: “D-Bound.” Highest Delta if it exceeds the next highest Delta by the target’s Delta-threshold.

For each run, the return from each algorithm should resemble what is shown in Table 8 which shows what was actually being run against and which In-Library target each algorithm would have returned.

Table 8. Algorithm responses to the run data for a set of parameters.

Run	Actual	Biggest_TLS	TLS_Bound	TLS & D	Big_D	D_Bound
1	Con1	T5	U	U	T1	U
2	T2	T2	T2	T2	T2	T2
3	Con2	T1	T1	T1	T1	T1
4	Con3	T2	T2	U	T1	U
5	Con4	T3	T3	T3	T3	T3
6	Con5	T1	T1	T1	T1	T1
7	Con6	T1	T1	T1	T1	T1
8	Con7	T3	T3	U	T2	U

The information present in the table is examined next. Table 8 shows Biggest_TLS and Big_D always return a target, as “U” is never shown in either of these algorithm’s columns. Also, Run 2 has each algorithm correctly identifying a T2 (Panzer) as a “T2” (“Panzer”). Run 1 is against confuser number 1 (Con1) and Run 3 is against confuser number 2 (Con2). Although each algorithm in Run 3 identifies the confuser as “T1” (“Clutter”), it is considered correct as “Clutter” is appropriate in this walkthrough to identify Out-Of-Library. Likewise, Run 1 has four algorithms that identify the confuser as either “Unknown” or “Clutter.” This walkthrough counts all four of these responses correct, since the confuser is Out-Of-Library.

It is mentioned that the algorithm responses are based on a given set of parameters. These parameters are found in the descriptions of each algorithm as the minimum threshold for TLS and as the minimum threshold for Delta. These values are not provided with the data file, and they can vary as needed. Thus, these values should be entered each time an analysis must be accomplished. Figure 9 and Figure 10 show how these values shall be entered. Once they are entered, the button “Done Entering Thresholds” may be pressed to spawn another MS Excel workbook with all the worksheets needed for the UIT.

Target	T1	Tiger	Panzer	T-72	Bradley
Min Likeness					
Min Delta					
Highest Poss. Population					
Most Likely Population					
Lowest Poss. Population					

Done Entering Thresholds

Figure 9. User-Defined Parameters Blank Sheet.

Target	T1	Tiger	Panzer	T-72	Bradley
Min Likeness	0.458	0.49	0.496	0.605	0.645
Min Delta	0.05	0.05	0.05	0.05	0.05
Highest Poss. Population	50	60	70	80	90
Most Likely Population	20	20	20	20	20
Lowest Poss. Population	10	10	10	10	10

Done Entering Thresholds

Figure 10. User-Defined Parameters Filled-in.

Although the objective is to find the best algorithm, each algorithm depends on the user-defined parameters. Essentially, this means each algorithm can be further improved or worsened by adjusting thresholds. Rather than just choosing the best algorithm of the five, the best solution should select a good algorithm and the best parameters for that algorithm.

Unfortunately, algorithms might be sensitive to not just one but all parameters. This is best shown using an example run. In the example, the TLS for “Clutter” is 0.7, the TLS for “Tiger” is 0.69 and the TLS response for every other target is 0.65. The TLS-threshold for each target will be 0.60. The Delta score is the difference between the TLS and the TLS-threshold. This case is shown in Table 9. We will also use

Delta-thresholds of 0.01. In this first case, “Clutter” would be the best response for all algorithms.

Table 9. An example run for TLS-thresholds.

	Type	Clutter	Panzer	Tiger	T-72	Bradley					
Case	TLS	0.70	0.69	0.65	0.65	0.65	Biggest TLS	TLS-Bound	TLS & D	Big D	D-Bound
	TLS-Threshold	0.60	0.60	0.60	0.60	0.60					
1	Delta	0.10	0.09	0.05	0.05	0.05	"Clutter"	"Clutter"	"Clutter"	"Clutter"	"Clutter"

As the environment changes, the TLS-thresholds may be redefined. For example, if clutter is unlikely, the user may redefine the TLS-threshold on Clutter to be higher. In case 2, the TLS-threshold is increased to 0.75. The new Delta scores are shown in Table 10.

Table 10. Second TLS-threshold case.

	Type	Clutter	Panzer	Tiger	T-72	Bradley					
Case	TLS	0.70	0.69	0.65	0.65	0.65	Biggest TLS	TLS-Bound	TLS & D	Big D	D-Bound
	TLS-Threshold	0.60	0.60	0.60	0.60	0.60					
1	Delta	0.10	0.09	0.05	0.05	0.05	"Clutter"	"Clutter"	"Clutter"	"Clutter"	"Clutter"
	TLS-Threshold	0.75	0.60	0.60	0.60	0.60					
2	Delta	-0.05	0.09	0.05	0.05	0.05	"Clutter"	"UNKNOWN"	"UNKNOWN"	"Panzer"	"Panzer"

In this case, “Unknown” is returned on two algorithms since the target with the highest TLS, “Clutter” does not meet the TLS-threshold. However, now “Panzer” has the biggest Delta, so the fourth algorithm returns “Panzer.” In addition, for the D-Bound algorithm, the highest Delta value must exceed the Delta-threshold or “Unknown” is returned. “Panzer’s” Delta of 0.09 exceeds the Delta-threshold of .01 set earlier, so “Panzer” is returned for the last algorithm as well.

Another case occurs if the TLS-threshold for a different target is lowered instead. In this case, shown as Case 3 in Table 11, the TLS-threshold for Bradleys is lowered to 0.4. This could happen if it is known that “Bradley” returns tend to be lower from the identification routine. In this case, “Bradley” would be returned for the

last two algorithms and “Unknown” is returned instead of “Clutter” for the third algorithm.

Table 11. Case 3 of TLS-Thresholds.

	Type	Clutter	Panzer	Tiger	T-72	Bradley					
Case	TLS	0.70	0.69	0.65	0.65	0.65	Biggest TLS	TLS-Bound	TLS & D	Big D	D-Bound
1	TLS-Threshold	0.60	0.60	0.60	0.60	0.60					
	Delta	0.10	0.09	0.05	0.05	0.05	"Clutter"	"Clutter"	"Clutter"	"Clutter"	"Clutter"
2	TLS-Threshold	0.75	0.60	0.60	0.60	0.60					
	Delta	-0.05	0.09	0.05	0.05	0.05	"Clutter"	"UNKNOWN"	"UNKNOWN"	"Panzer"	"Panzer"
3	TLS-Threshold	0.60	0.60	0.60	0.60	0.40					
	Delta	0.10	0.09	0.05	0.05	0.25	"Clutter"	"Clutter"	"UNKNOWN"	"Bradley"	"Bradley"

Switching any one threshold, in this example, the TLS-threshold on “Clutter” from 0.6 to 0.75 or the TLS-threshold on “Bradley” from 0.6 to 0.4, can cause multiple changes in all algorithm returns. Although this example may seem unlikely, some of the test runs already encountered had similar results. For instance, Test Run #420 of the sample data had the results shown in Table 12.

Table 12. Test Run #420 results.

Actual	"Clutter"	"Tiger"	"Panzer"	"T-72"	"Bradley"
Panzer	0.689	0.69	0.648	0.468	0.689

In this case, the return values for “Clutter,” “Tiger” and “Bradley” are 0.689 or 0.69. The correct return of “Panzer” only has a TLS of 0.648, so TLS-based algorithms would have to choose between three wrong choices with “Tiger” only edging the other two out by 0.001. The determination of Delta-based algorithms, however, depends not only on the TLS-threshold for “Panzer,” but on all thresholds. For example, the TLS-thresholds, like those chosen in Table 13, would make four of five Delta scores even.

Table 13. Run #420 with TLS-thresholds altered and Delta scores calculated.

Actual	"Clutter"	"Tiger"	"Panzer"	"T-72"	"Bradley"
Panzer	0.689	0.690	0.648	0.468	0.689
Min-Bound	0.641	0.642	0.600	0.500	0.641
Delta Scores	0.048	0.048	0.048	-0.032	0.048

The sensitivity of each algorithm to not just one threshold but to each threshold shows that target declaration is dependent on multiple parameters. Varying just one threshold may not cause a change from a correct identification to a misidentification, but possibly a misidentification to another misidentification or to a declaration of "Unknown." This topic will be further reviewed in the section on ROC curves.

Algorithm Calculations

Sensitivity to thresholds shows a need to be able to adjust the calculations in the UIT to varying thresholds. With User-Defined parameters being uncertain and flexible, the UIT should be able to handle a robust range of thresholds. Having a static analysis where a button is pressed and results were calculated for one set of user-defined parameters was impractical. However, a dynamic system, where all results could change as a user enters new parameters would allow the user to see changes as they were made.

To create a dynamic system, all calculations are made by reference. In MS Excel, using active references and formulas rather than values resulting from VBA macros allows calculations to be updated as any information changes. Calculations would have to be broken down into a way allowing VBA to generate solutions only through references. To accomplish this, each algorithm is broken down into necessary elements that would use references and yet still be easy for a macro to form. The first necessary elements are Delta scores.

A Delta score, as shown in our previous example, is the difference between the TLS score and the TLS-threshold. For our sample data, the TLS-thresholds used are given in Figure 10. In Figure 11, these same values will be placed above the target declarations in which they correspond. The Delta-thresholds were also given in Figure 10. In Figure 11, they also reside above the Delta values in which they correspond. In the figure, D1 through D5 refer to the Delta scores of the five targets. These names are generic, since they shall be created by the VBA macro.

Bound	0.458	0.490	0.496	0.605	0.645	0.05	0.05	0.05	0.05	0.05
Actual	"Clutter"	"Tiger"	"Panzer"	"T-72"	"Bradley"	D1	D2	D3	D4	D5
Con1	0.513	0.517	0.525	0.393	0.601	0.055	0.027	0.029	-0.212	-0.044
T2	0.48	0.596	0.421	0.372	0.457	0.022	0.106	-0.075	-0.233	-0.188
Con2	0.691	0.467	0.528	0.371	0.512	0.233	-0.023	0.032	-0.234	-0.133
Con3	0.632	0.651	0.519	0.502	0.53	0.174	0.161	0.023	-0.103	-0.115
Con4	0.477	0.445	0.605	0.436	0.447	0.019	-0.045	0.109	-0.169	-0.198

Figure 11. Placement of data for algorithm calculations.

To create the Delta values, the macro need only reference any value given in the data and subtract the TLS-threshold. By placing the TLS-thresholds above the target names, any value need only subtract its value from a number always in the same row of its column. Delta scores are then easily created with two references as long as the data and the TLS-thresholds are valid. Whereas a TLS ranges from zero to one, Delta scores can range from negative one to one. A negative one can occur when the TLS is zero and the TLS-threshold is one. This case is unlikely, but possible.

The TLS-thresholds above the target names are not values but references. They reference threshold values that have been recreated on another worksheet entitled, "Algorithms." The "Algorithms" worksheet is intended for the user's main point of interface. It includes Vertical and Horizontal statistics and their corresponding ROC curves. Placing the thresholds on this page allows the user to enter new thresholds and see the end results. All calculation worksheets have references to the values on this

page. A snapshot of this page is included as Figure 11. A close-up picture of the area to change user-defined parameters is shown in Figure 12.

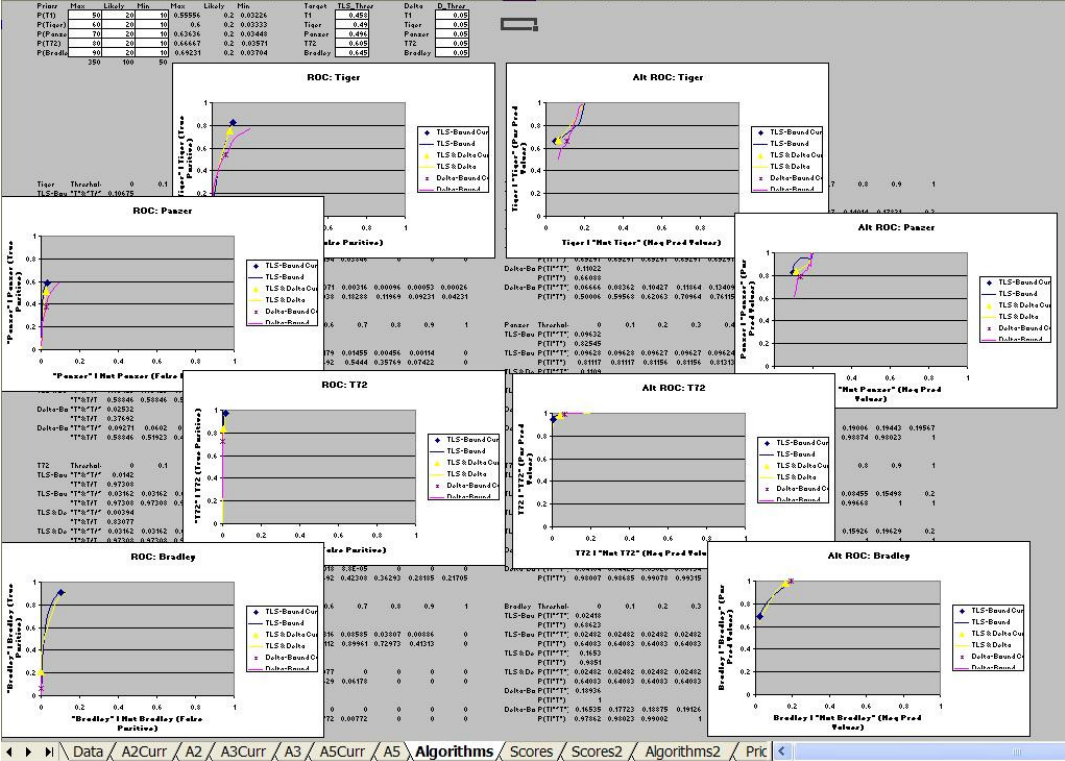


Figure 12. Snapshot of the entire “Algorithms” worksheet.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1		Priors	Max	Likely	Min	Max	Likely	Min		Target	TLS Thres		Delta	D Thres	
2		P(T1)	50	20	10	0.555556	0.2	0.032258		T1	0.458		T1	0.05	
3		P(Tiger)	60	20	10	0.6	0.2	0.033333		Tiger	0.49		Tiger	0.05	
4		P(Panzer)	70	20	10	0.636364	0.2	0.034483		Panzer	0.496		Panzer	0.05	
5		P(T72)	80	20	10	0.666667	0.2	0.035714		T72	0.605		T72	0.05	
6		P(Bradley)	90	20	10	0.692308	0.2	0.037037		Bradley	0.645		Bradley	0.05	
7			350	100	50										

Figure 13. Area for user to change user-defined parameters.

User-defined input is changeable, so other worksheets reference these values from the “Algorithm” worksheet for their calculations. User-defined values are entered in white cells to indicate a user can change these values as desired. These values are Priors, TLS-Thresholds, and Delta-Thresholds. The Priors are not probabilities but the number of units expected to be encountered in the given operating

environment. The probabilities from these populations are calculated in grey next to the white boxes. The values in grey are calculations references to or referenced by other pages and should not be manually changed. If the values are manually changed by mistake, it is suggested that the file be spawned again from the master spreadsheet.

The Priors are entered as populations, but prior probabilities are needed for actual calculations. Three probabilities are calculated: Max, Likely, and Min. Likely is calculated by adding the Likely populations of all targets. Then, the target's Likely population is divided by the sum to give the probability of when a target is encountered, it is that target. The Max probability uses the sum of the maximum population of a target and the minimum populations of each of the other targets. Then, the maximum population is divided by this sum. The result is the maximum probability possible for encountering that target when a target is encountered. A similar calculation is used for Min.

Populations are originally used rather than probabilities for two reasons. First, the sum of all Likely probabilities must add to one. Rather than error checking, a formula ensures these probabilities add to one. Max and Min do not follow this rule, since each of these probabilities use different sets of populations. Finally, estimating number of units remaining should be easier than estimating probabilities. The priors in Figure 13 could come from a decision-maker estimating that at the very least 10 Panzers are left in the operational environment or at the very least it should be 3.4483% likely to encounter a Panzer when a target is encountered. The first case seems easier to estimate.

Next, maximum values must be calculated. By referencing a row of TLS scores, the maximum TLS can be found. Similarly, a maximum Delta score is found

for all Delta values in each row. Once these are determined, algorithm responses for Biggest-TLS and Big-D are if-statements using the maximum value as criteria and finding which column has that value. Using the run example used in Table 12 and the actual thresholds, the Biggest-TLS and Big-D responses were calculated similar to the way shown in Table 14 and Table 15 respectively. In the case of a tie, as stated in the assumptions, the first target encountered in the tie has priority and will be returned.

Table 14. Biggest-TLS Calculation.

Actual	"Clutter"	"Tiger"	"Panzer"	"T-72"	"Bradley"	Max-TLS	Biggest-TLS
Panzer	0.689	0.690	0.648	0.468	0.689	0.69	"Tiger"

Table 15. Biggest-D Calculation.

Actual	"Clutter"	"Tiger"	"Panzer"	"T-72"	"Bradley"	Max-D	Big-D
Panzer	0.231	0.200	0.152	-0.137	0.044	0.231	"Clutter"

The algorithms, TLS-Bound and D-Bound should return either that same response as Biggest-TLS and Big-D, respectively, or they should return "Unknown" if the TLS-threshold or Delta-threshold is not met. However, which of the thresholds to use is not readily apparent at this point, since the threshold is in a column determined by the solution to the previous two algorithms. Thus, these algorithms must use the information determined previously in order to be calculated. For example, TLS-Bound should use the threshold in the column with "Tiger," and D-Bound should use the threshold in the column with "Clutter." If the user enters new thresholds, the solutions to the previous algorithms could change, in turn, changing which column to be used. Thus, a long if-statement reference is required to keep the spreadsheet dynamic. The if-statement chooses a column based on the response to the previous solution. Since the previous solution is dynamic, these solutions remain dynamic.

To compute the final algorithm, “TLS & D,” another piece of information is necessary. “TLS & D” returns the same response as TLS-Bound as long as the target being declared has the largest Delta as well as the largest TLS. This can be determined in one of two ways. One way is to calculate the Delta of the highest TLS value and compare it to the maximum Delta. The other is to compare the results of TLS-Bound with Big-D, and if they match, declare the same response. Although the second method is easier, it can fail in cases of a tie. Ties are broken based on priority, but if a lower priority target has a higher TLS and an equal Delta with a higher priority target, “TLS & D” should still declare that target the response rather than “Unknown.” Thus, the Delta of the highest TLS-scoring category is added as another column entitled “Delta” for the Delta of the current choice. The “TLS & D” response for the example is shown in Table 16.

Table 16. “TLS & D” Calculations.

	"Clutter"	"Panzer"	"Tiger"	"T-72"	"Bradley"	Max-TLS	Max-D	Delta	Biggest TLS	TLS & D
TLS	0.70	0.69	0.65	0.65	0.65	0.690			"Tiger"	
TLS-Threshold	0.60	0.60	0.60	0.60	0.60					
Delta	0.10	0.09	0.05	0.05	0.05		0.231	0.200		"Unknown"

Only a limited number of columns are necessary to calculate all algorithms. One column is necessary for each algorithm. For each In-Library target, one column is needed for the TLS and one for the Delta score. Next, a column is needed to list which TLS-threshold and which Delta-threshold is being used. Then, a column is needed for the maximum TLS of a row, the maximum Delta of a row, and the Delta of the target with the biggest TLS. Thus, if k is the number of In-Library targets, $2k+5+5$ columns are needed. One additional column indicating the actual target in the run is unnecessary for the algorithm calculations, but it is added for later use and current

insight. Later to calculate values for the ROC and Vertical ROC curves, a few more columns are necessary. They are necessary to determine the next highest Delta score, entitled “Other_D,” and the difference between the two, entitled “Beat_Delta.” These columns increase the number of columns needed to $3k+5+5+2$. The number of columns might become important later, as MS Excel has a limited number of columns per worksheet for use.

ROC Curves

The ROC curves generated for this data are not the usual ROC curves. Usual ROC curves deal with two declarations and one threshold. Changing the threshold determines whether a “TRUE” or “FALSE” is declared. Lower the threshold enough, and all samples are called “TRUE,” while raising it should eventually have all samples declared “FALSE.” True Positives, the probability of “TRUE” given that it’s TRUE, is plotted against False Positive, the probability of “TRUE” given that it’s FALSE. If the threshold is low enough that everything is declared “TRUE” then values for both the True Positive and False Positive are 1:

$$P(\text{“TRUE”}|\text{TRUE}) = 1 \text{ and the } P(\text{“TRUE”}|\text{FALSE}) = 1.$$

If the threshold is high enough, then nothing is declared “TRUE” and values for both the True Positive and False Positive are 0:

$$P(\text{“TRUE”}|\text{TRUE}) = 0 \text{ and the } P(\text{“TRUE”}|\text{FALSE}) = 0.$$

This is not the case with the sample data set.

The sample data set has multiple responses and either no thresholds or multiple thresholds. The algorithms with no thresholds, Biggest-TLS and Big-D, will always return one response, the In-Library target with the biggest TLS or the biggest Delta, respectively, regardless of the thresholds.

Also, as shown above in the section on User-Defined Parameters, lowering the threshold for an In-Library target does not guarantee that the corresponding declaration will ever be returned. Declarations are dependent on all thresholds and all TLS. Dependence on TLS is easily shown with the Biggest-TLS, TLS-Bound, and the “TLS & D” algorithms, since the only response ever considered by these algorithms is either the declaration with the highest TLS or “Unknown.” If the correct response did not have the highest TLS, it will never be considered as an answer for these three algorithms. Thus, any ROC curve may not have a threshold value where the True Positive and False Positive equal 1. If the threshold is increased, then – except for the Biggest-TLS algorithm, Big-D algorithm, and the case where Clutter is the actual target – there should be a threshold where the True Positive and False Positive equal 0. This means any ROC curve created may have an upper limit other than one on both True Positives and False Positives.

Clutter is a special case. In the case where the Clutter is the actual target, lowering the threshold on “Clutter” might change Delta values to increase the number of “Unknowns” declared in some algorithms, but it does not guarantee all samples being declared “Clutter.” Raising the threshold for Clutter declarations, on the other hand, might change declarations from “Clutter” to “Unknown.” However, “Unknown” is still an acceptable response for Clutter. Thus, raising the threshold does not guarantee all “Clutter” and “Unknown” responses will be eliminated. Thus, Clutter is not considered for a ROC curve plot. The threshold for “Clutter” will, however, affect the plots for the other declarations.

Multiple declarations also require a modification to the ROC curves. Each declaration is competing against every other declaration in the library as well as

“Unknown.” To create a two-dimensional ROC curve, each In-Library target must be considered separately. For each target’s case, a “TRUE” will indicate the algorithm chose that target for its declaration whereas, “FALSE” will indicate that some other target was declared. Thus, a ROC curve must be plotted for each target. In addition, a ROC curve can be plotted for each algorithm. Since algorithms are constrained to at most three curves as two of the five are unaffected by thresholds, each graph in the UIT shall plot all three algorithms in one graph for each individual In-Library target excluding Clutter.

Although Clutter will not be plotted as its own ROC curve, it will still be used for the other ROC curves. For each target, “Clutter” and “Unknown” will be counted as a declaration of “FALSE.” Although it may be suggested that “Unknowns” be taken out of the consideration, the thresholds being used can change a declaration to “Unknown” instead of “FALSE,” so “Unknown” shall be considered a “FALSE” case.

ROC Curve Calculations

After considering how ROC curves will handle multiple thresholds and multiple declarations, being able to build a robust, routine, and dynamic ROC curve is the next challenge. Although the ROC curve for each algorithm will be plotted together on one graph for every target, it still implies a ROC curve is necessary for every non-Clutter In-Library target and three algorithms. Considerations must be made for the calculations on the limitations of MS Excel as well.

To determine ROC curves, one worksheet is allocated to each ROC-friendly algorithm: TLS-Bound, D-Bound, and “TLS & D”. These worksheets are named A2, A5, and A3 respectively. In addition, three additional worksheets were made so that the current operating point on the ROC curve for the algorithm would be apparent.

Each worksheet will have calculations for every target. For calculations to be made dynamically, the range of thresholds must be consistent, even if the thresholds change. Thus, thresholds are plotted at 0.1 increments from 0 to 1. It should be noted that a TLS-Bound with a 0 threshold is the same as Biggest-TLS and D-Bound with a -1 threshold is the same as Big-D since 0 is the lowest TLS value and -1 is the lowest Delta value. At each threshold, a point on the ROC curve shall be plotted. Next, we shall examine what values are needed to calculate points for each algorithm.

For TLS-Bound, only a few additional columns are necessary. Since a True Positive is $P(\text{"TRUE"}|\text{TRUE})$, we are interested in indicating if a row is TRUE and whether the response at each threshold is "TRUE." With Unknowns being counted as FALSE, FALSE only happens when it is not TRUE or $\sim\text{TRUE}$, so one column will suffice to indicate if a row is TRUE or FALSE. In practice, the UIT will use a binary number to indicate whether a row is TRUE or FALSE for each target.

Although the actual target never changes due to changes in threshold, the declarations change with threshold. Thus, one column is needed per target per threshold. However, TLS-Bound and TLS & D algorithms only have two declarations available. These declarations are "Unknown" or the same declaration as Biggest TLS. Since this is true regardless of the thresholds, another binary indication of whether the declaration of "TRUE" was made under Biggest-TLS is included. Since "Unknowns" are considered "FALSE," only two responses for this column are allowed: "FALSE" and "TRUE." These responses are converted into 0 and 1, respectively. Each threshold can use this declaration column to help calculate its own response.

For each threshold, rather than plotting "TRUE" and "FALSE" in each column, the information was combined with TRUE and FALSE to save room. Thus,

“TRUE”|TRUE is indicated by an “A”, “FALSE”|TRUE is indicated by a “B”,
“TRUE”|FALSE is indicated by a “C”, and “FALSE”|FALSE is indicated by a “D.” A
sample of the resulting table is shown in Table 17. Its placement of the worksheet is
shown in Figure 14.

Table 17. Truth data calculations.

T1	~T1	"T1"	0	0.1	0.2	0.3	0.4
1	0	0	C	C	C	C	C
1	0	1	A	A	A	A	A
1	0	0	C	C	C	C	C
1	0	0	C	C	C	C	C
1	0	1	A	A	A	A	A
1	0	1	A	A	A	A	A

		A	P(T)	0.2					
		B	P(~T)	0.8					
		Threshold		0	0.1	0.2	0.3	0.4	0.5
D	P("T" ~T)	"T"&~T/~T	0.060595	0.060595	0.060595	0.060595	0.060595	0.060595	0.060595
C	P("T" T)	"T"&T/T	0.250175	0.250175	0.250175	0.250175	0.250175	0.2487	0.2487
		P("T")	AC + BD	0.098511	0.098511	0.098511	0.098511	0.0982	0.0982
		P("T"&T)	AC	0.050035	0.050035	0.050035	0.050035	0.0497	0.0497
		P("T"&~T)	BD	0.048476	0.048476	0.048476	0.048476	0.0484	0.0484
		P(~T"&T)	A*(1-C)	0.149965	0.149965	0.149965	0.149965	0.1502	0.1502
		P(~T"&~T)	B*(1-D)	0.751524	0.751524	0.751524	0.751524	0.7515	0.7515
		P(T ~T)		0.166353	0.166353	0.166353	0.166353	0.1666	0.1666
		P(T T)	AC/(AC+B)	0.507911	0.507911	0.507911	0.507911	0.5065	0.5065
0.05	0.05								
T72	Bradley	T1	Tiger	Panzer	T72	Bradley	T1	~T1	"T1"
-0.212	-0.044	-1	0.027	0.029	-0.212	-0.044	1	0	0 C
-0.234	-0.133	-1	-0.023	0.032	-0.234	-0.133	1	0	1 A
-0.103	-0.115	-1	0.161	0.023	-0.103	-0.115	1	0	0 C
-0.169	-0.198	0.019	-0.045	-1	-0.169	-0.198	1	0	0 C
-0.208	-0.137	-1	0.196	0.1	-0.208	-0.137	1	0	1 A
-0.119	-0.043	-1	0.044	-0.039	-0.119	-0.043	1	0	1 A

Figure 14. Truth data calculations spreadsheet screenshot.

In this figure, the column entitled T1 shows a 1 if the actual target is a T1
(Clutter). Thus, T1 is a column for TRUE in this case. Likewise, ~ T1 is a column
for FALSE. The next column, “T1” indicates a declaration of “T1” in the Biggest TLS
algorithm. This sample was taken from the A2 worksheet, signifying TLS-Bound
results. Since the algorithm Biggest-TLS is the special case of TLS-Bound with a

threshold of 0, the column where the threshold is 0 is actually is the declaration that would be made in Biggest-TLS, so it should be the same as the “T1” column except with TRUE and FALSE considered. The ~T1 column was found unnecessary, but it remains for error-checking and as a space holder due to the spacing needed for the other calculations shown above the rows that have not yet been discussed.

As stated earlier, the columns after “T1” are labeled by the threshold to which they correspond. Underneath each threshold is a letter corresponding to a block in a probability matrix. The probability matrix and its coding are shown in Table 18.

Table 18. Probability Matrix.

	"T"	"~T"
T	A	C
~T	B	D

To determine which letter is correct involves an if-statement. A sample if-statement is as follows:

```
=IF(AND($AO24=1,$AM24=1,$H24>AT$19),"A",
IF(AND($AO24=1,$H24>AT$19,$AM24=0),"B",
IF(AND(OR($AO24=0,$H24<AT$19),$AM24=1),"C",
IF(AND(OR($AO24=0,$H24<AT$19),$AM24=0),"D","E"))))
```

The first statements in the AND() function verify that it is TRUE and “TRUE.” The last statement in the AND() function determines if the value of the highest TLS, the “Max” column computed earlier, is greater than the TLS-threshold identified in the “Thres” column. If everything is true, then an “A” is returned. The rest of the if-statement does likewise comparisons and determines to which part of the probability matrix the value corresponds. A final value of “E” is included to identify any mistakes

and ensure that a coded letter is chosen based on their corresponding statement being true. No “E’s” were found in the analysis of the sample data.

D-Bound’s information is calculated in much the same way as TLS-Bound was. The difference is Big-D is used instead of Biggest-TLS in the declaration column. A declaration column is shown in the example as the column with the “T1” heading.

“TLS & D” is slightly different. Like TLS-Bound, it uses the information gathered from Biggest-TLS. It cannot use the information gathered from TLS-Bound, since TLS-Bound depends on the TLS-threshold. Thus, it must make the same check as done for TLS-Bound. In addition, it must calculate a new Delta value for comparison with each new threshold. This value must be compared to the maximum Delta value of the other targets.

Since the maximum Delta of the other targets must be identified, a column is included entitled “Other_D.” Although it seems this must be replicated for each target, the UIT takes advantage of “TLS & D’s” reliance on the value with the biggest TLS. As it has already been stated, the only outcomes of “TLS & D” are the value of the biggest TLS or “Unknown.” Thus, Other_D only needs to be calculated for the target identified in Biggest-TLS. The example used earlier in Table 16 can illustrate this point. Thus, the table is shown again here as Table 19.

Table 19. Example for Alt_D.

Actual	"Clutter"	"Tiger"	"Panzer"	"T.72"	"Bradley"	Max-TLS	Max-D	Curr-D	Biggest-TLS	TLS & Delta
Panzer	0.689	0.690	0.648	0.468	0.689	0.69			"Tiger"	
Min-Bound	0.458	0.490	0.496	0.605	0.645					
Delta Scores	0.231	0.200	0.152	-0.137	0.044		0.231	0.2		"Unknown"

In this example, Biggest-TLS’s response is “Tiger.” Even though the actual target is a Panzer, “Tiger” had the biggest TLS. For any In-Library target other than

“Tiger,” the only possible responses are “Tiger” and “Unknown” which are both “FALSE.” For Tiger under “TLS & D,” the possible responses are “Tiger” and “Unknown,” which correspond to “TRUE” and “FALSE.” These declarations are dependent on the thresholds. Thus, only for Tiger will an alternate Delta need to be calculated. In this case, the maximum Delta for the other targets, called Other_D, is 0.231. If the threshold for “Tiger” was lowered by 0.032, the response for “TLS & D” would change from “Unknown” to “Tiger.” In the probability matrix, the coding would change from D to B indicating the change from “FALSE”|FALSE to “TRUE”|FALSE since the actual target was a Panzer.

Only one Other_D needs to be calculated. To do this, another column is created for each Delta score. The Delta score of the target identified in Biggest-TLS is reduced to -1. Then, the maximum of the Delta scores is calculated. Since -1 is the lowest possible Delta score, the maximum of all these new Delta scores is the maximum of the Delta scores not identified in Biggest-TLS. The value 0 was not used in this case, since it is possible to have each Delta score negative. Once this is accomplished, the if-statement to determine which letter should be used per threshold per target is:

```
=IF(AND($AG24=1,$AE24=1, $I24> AK$22,$I24-AK$22>=$J24),"A",
IF(AND($AG24=1,$I24>AK$22,$I24-AK$22>=$J24,$AE24=0),"B",
IF(AND(OR($AG24=0,$I24<AK$22,$I24-AK$22<$J24),$AE24=1),"C",
IF(AND(OR($AG24=0,$I24<AK$22,$I24-AK$22<$J24),$AE24=0),"D",
"E"))))
```

Along with the comparisons made for TLS-Bound, the if-statement compares the Max to the new threshold to create a new Delta score. This score is then compared to the

ROC Curve Points and Plots

Once the truth data is compiled, summary statistics must be calculated for plots to be created. Figure 13, shown again here as Figure 14, shows where the summary data is positioned on each worksheet for ease of calculations.

[illegible]

Figure 15. Positioning of Truth summary data.

For most ROC curves, the priors do not play a role. Instead, all that is needed is if a run resulted in a “TRUE” or not and whether that run was based on a TRUE case or a FALSE case. Due to classifying each run as one of four cases in a probability matrix, this information is readily available. By using COUNTIF() functions, summary data from all each column can be calculated and placed into two cells. One cell shows the True Positive result for the column, $P(\text{“TRUE”}|\text{TRUE})$, and is calculated by:

=COUNTIF(AV24:AV3916,"B")/
(COUNTIF(AV24:AV3916,"B")+COUNTIF(AV24:AV3916,"D"))

The other cell shows the False Positive results for the column, P("TRUE"|FALSE), and could be calculated by:

$$=IF((COUNTIF(AV24:AV3916,"A")+COUNTIF(AV24:AV3916,"C"))=0, 0, COUNTIF(AV24:AV3916,"A")/(COUNTIF(AV24:AV3916,"A")+COUNTIF(AV24:AV3916,"C")))$$

The if-statement was added to prevent divisions by zero.

Unfortunately, this is not a typical ROC curve since FALSE can occur by various means. The P("TRUE"|FALSE) actually depends on what target FALSE is. In other words, P("TRUE"|FALSE) in the case of a Panzer can be written P("Panzer"|Not a Panzer). If a Tiger is always confused as a Panzer, but a Bradley is never confused as a Tiger, then the P("Panzer"|Not a Panzer) will depend on how many Tigers and Bradleys will be encountered. If only Tigers remain, this probability will be one. If only Bradleys remain, the probability will be zero.

To determine the correct P("TRUE"|FALSE), Bayes' Rule must be used (Wackerly, 2002: 68). For example, consider the case where the target in question is Tiger or T2. The desired probability is P("Tiger"|Not a Tiger) or P("T2" | ~T2). The options for Not a Tiger or FALSE are each of the other targets: Clutter, Panzer, Bradley, or T-72. These cases are mutually exclusive as one target is recognized at a time.

Thus, the Bayes' Rule for the target T2 can be written as:

$$\begin{aligned} & P("T2"| \sim T2) \\ &= \frac{P(\sim T2 \cap "T2")}{P(\sim T2)} \\ &= \frac{P((T1 \cup T3 \cup T4 \cup T5) \cap "T2")}{P(\sim T2)} \\ &= \frac{P(T1 \cap "T2") + P(T3 \cap "T2") + P(T4 \cap "T2") + P(T5 \cap "T2")}{P(\sim T2)} \\ &= \frac{P(T1) * P("T2"|T1) + P(T3) * P("T2"|T3) + P(T4) * P("T2"|T4) + P(T5) * P("T2"|T5)}{P(\sim T2)} \end{aligned} \tag{2}$$

To calculate this value in MS Excel each of these probabilities must be identified and referenced. P(T1) through P(T5) are already given as prior probabilities. P(~T2) is simply one minus P(T2). To obtain the other probabilities, the conditional probabilities, the data in MS Excel is sorted before any calculations begin. The data is sorted by the priority of In-Library targets. Then the Countif() function used earlier only counts only rows corresponding to each individual target. Thus, the P("Declaration"|Specific Target) can be obtained by counting responses in each target's section. The summation occurs underneath the algorithm calculations and is referenced by the summary cell for P("TRUE"|TRUE). A picture of where this calculation takes place is given in Figure 16.

0.3	0.332	-0.001	-1	0	1	0 D	D	D	D
0.208	0.328	-0.077	-1	0	1	0 D	D	D	D
0.254	0.283	0.085	-1	0	1	0 D	D	D	D
P(T1)				0.2 T1		0	0	0	0
P(Tiger)				0.2 Tiger		0.01	0.01	0.01	0.01
P(Panzer)				0.2 Panzer		0.010526	0.010526	0.010526	0.010526
P(T-72)				0.2 T-72		0	0	0	0
P(Bradley)				0.2 Bradley		0	0	0	0

Figure 16. Bayes' Rule Applied.

Once the summary data is calculated, data can be compiled to make the ROC curve. Data from each algorithm worksheet – A2, A3, and A5 – is grouped together by target on a new worksheet, Algorithms. The table of algorithms for one target could look like Table 20. A graph from the table could look like Figure 17.

Table 20. Table of Algorithms for one target.

Tiger	Threshold	0.1	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
TLS-Bound	T & ~T / ~T	0.19433	0.19433	0.19433	0.19433	0.19433	0.18309	0.15808	0.1097	0.05257	0.01295	0
	T & T / T	0.82308	0.82308	0.82308	0.82308	0.82308	0.82308	0.80385	0.64729	0.36538	0.13846	0
TLS & Delta	T & ~T / ~T	0.19433	0.19433	0.19433	0.19433	0.19433	0.12992	0.01624	0.00028	0	0	0
	T & T / T	0.82308	0.82308	0.82308	0.82308	0.82308	0.70769	0.26154	0.03846	0	0	0
Delta-Bound	T & ~T / ~T	0.26094	0.20055	0.12941	0.05594	0.01459	0	0	0	0	0	0
	T & T / T	0.76923	0.76154	0.64479	0.37066	0.15	0.00385	0	0	0	0	0

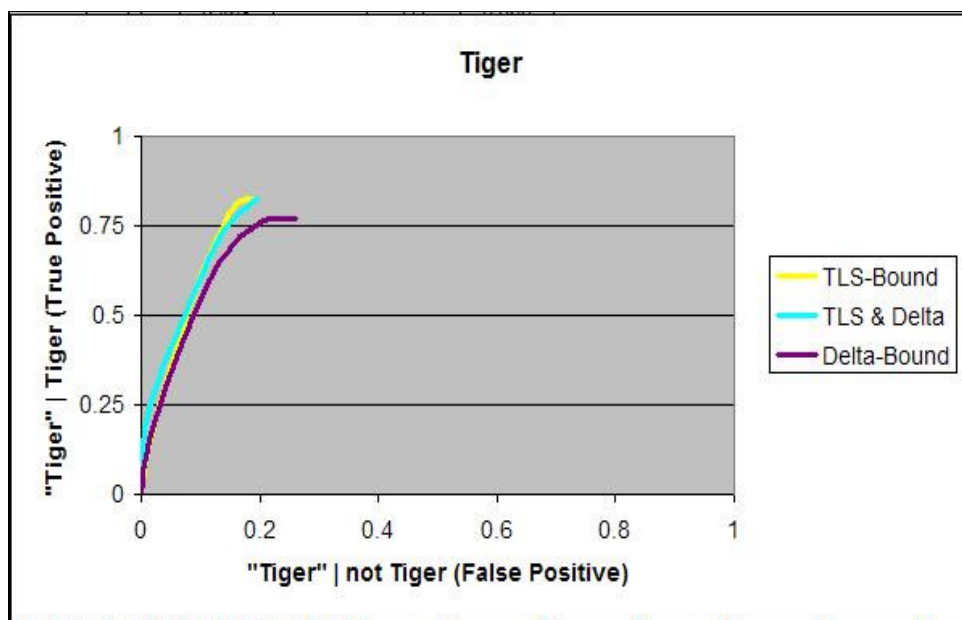


Figure 17. ROC curve for one target.

Figure 17 shows how an upper bound exists for both the True Positive and the False Positive. The upper bound is caused by this ROC curve's reliance on the highest TLS or the highest Delta depending on the algorithm. For example, if a Tiger is recognized, but the TLS is 0.5 for "Tiger" and 0.6 for "Panzer," lowering the threshold for "Tiger" will never result in "Tiger" being the highest TLS. Depending on the threshold for "Panzer," the return will either be "Panzer" or "Unknown;" both of which are misidentifications. Changing thresholds only matter when "Tiger" is the highest TLS or when determining Delta scores. Thus, even at the lowest thresholds, "Tiger" may not have a 100% declaration rate. This result is discussed in more detail in Section IV, Results and Analysis.

Figure 17 does not show where the current operating conditions are. For MS Excel to plot current operating points, new sets of data needed to be added. Thus, new worksheets were created – A2Curr, A3Curr and A5Curr – to calculate the current

position of each algorithm. In each of these worksheets, rather than testing over a range of thresholds, the test was confined to only the current threshold. This threshold is still referenced to the “Algorithms” spreadsheet, so any change would instantly change end results. The resulting table for the Tiger target is in Table 21 while a sample graph is shown as Figure 18.

Table 21. Table of Algorithms for one target with Current Thresholds.

Tiger	Threshold	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
TLS-Bounc	"T"&~T/~T	0.106752										
	"T"&T/T	0.823077										
TLS-Bounc	"T"&~T/~T	0.10936	0.10936	0.10936	0.10936	0.10936	0.106051	0.096258	0.071585	0.026542	0.006753	0
	"T"&T/T	0.823077	0.823077	0.823077	0.823077	0.823077	0.823077	0.807692	0.651163	0.365385	0.138462	0
TLS & Del	"T"&~T/~T	0.092388										
	"T"&T/T	0.75										
TLS & Del	"T"&~T/~T	0.10936	0.10936	0.10936	0.10936	0.10936	0.081513	0.009549	8.76E-05	0	0	0
	"T"&T/T	0.823077	0.823077	0.823077	0.823077	0.823077	0.707692	0.261538	0.038462	0	0	0
Delta-Bour	"T"&~T/~T	0.069113										
	"T"&T/T	0.53876										
Delta-Bour	"T"&~T/~T	0.192263	0.104284	0.069113	0.033305	0.018323	0.004558	0.002367	0.000701	0.000263	8.76E-05	0
	"T"&T/T	0.769231	0.633205	0.53876	0.420849	0.316602	0.219231	0.135135	0.1	0.042308	0.019231	0

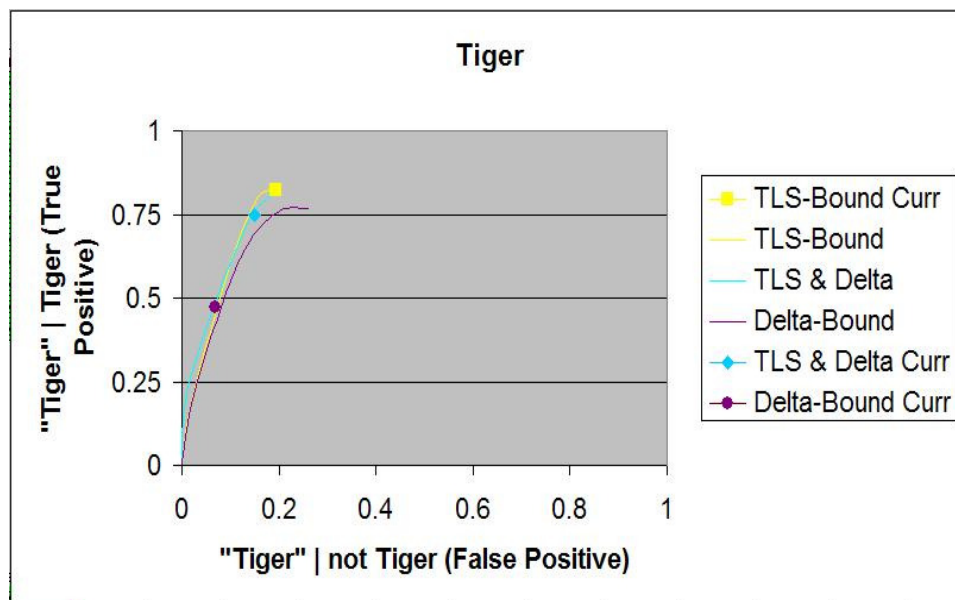


Figure 18. ROC curve with Current Thresholds.

Figure 18 shows how the current operating point may be a little off of the ROC curve since the ROC curve is only plotted at 0.1 increments. Since a curved line is fitted, it may also look to be curving down at some points. However, the ROC curve is

still useful. This plot shows that in this example the TLS-threshold for “Tiger” is good for TLS-Bound and TLS & D. However, the current True Positive value for D-Bound is low. By trading for poorer performance in False Positives, a much higher True Positive can be obtained for the D-Bound algorithm by reducing the Delta-threshold. This may be surprising as the Delta-threshold is set at 0.05. However, since some runs can have all negative Delta scores, a lower Delta-threshold is not unheard of. Due to the low nature of Delta scores, the thresholds used to generate points for D-Bound’s ROC curve has actually been changed from 0 to 1 in 0.1 increments to a set of thresholds better suited for the sample data. Due to the size limitations of MS Excel, only a set of eleven numbers is used. Agreeing upon the best set of thresholds to use may be a topic for future study.

Vertical ROC Curves

The ROC curve above shows a systems approach to error. If the ROC curve indicates poor performance, the system must be improved so that the correct target will be declared given it is the actual target. As stated in Section II, Literature Review, this is considered a Horizontal MOP. However, an operator is more concerned with whether a target is what the system declares it is as well as the probability a target goes undeclared or is misidentified as another target. In other words, the operator is more concerned about whether the enemy tank on the radar screen is really an enemy tank, $P(\text{TRUE} | \text{TRUE})$ and whether Surface-to-Air Missiles, in general, aren’t being identified as “Unknown,” “Clutter,” or friendly vehicles, $P(\text{TRUE} | \text{FALSE})$. These occurrences are examples of Vertical MOPs.

Rather than calculating Vertical MOPs for each target and each algorithm at a low, current and high threshold level, another option is to graph all Vertical MOPs

much like a ROC curve. A ROC curve plots the Horizontal MOP values, True Positives, $P(\text{"TRUE"}|\text{TRUE})$, and False Positives, $P(\text{"TRUE"}|\text{FALSE})$. An alternative ROC curve, referred to in this thesis as a Vertical ROC curve, could plot Prior Predictive Values, $P(\text{TRUE}|\text{"TRUE"})$ and Negative Predictive Values, $P(\text{TRUE}|\text{"FALSE"})$.

The Vertical ROC curves are not true ROC curves. As a true ROC curve uses the $P(\text{"TRUE"}|\text{TRUE})$, it can be calculated as $P(\text{"TRUE"}\&\text{TRUE}) / P(\text{TRUE})$. Here, the divisor, $P(\text{TRUE})$, is stable and does not depend on the threshold. However, the Positive Predictive Value, $P(\text{TRUE}|\text{"TRUE"})$, can be calculated as $P(\text{"TRUE"}\&\text{TRUE}) / P(\text{"TRUE"})$, where $P(\text{"TRUE"})$ definitely depends on the threshold. Even if the threshold is lowered to its minimum, the $P(\text{"TRUE"})$ might still be zero if the sample size is small or the return for the target is faulty. $P(\text{"TRUE"}\&\text{TRUE})$ will always be less than $P(\text{"TRUE"})$, but it can vary wildly as $P(\text{"TRUE"})$ changes with threshold. When $P(\text{"TRUE"})$ is zero, a one shall be used for $P(\text{TRUE}|\text{"TRUE"})$. This assumes that at the highest threshold, only a TRUE target would make that threshold. It shall be shown later that if this assumption is faulty, it will be apparent in the Vertical ROC curve. However, in practice as far as this research has gone, this is a valid assumption.

The Negative Predictive Value also has an interesting affect at the extreme threshold. If the threshold is increased to its upper limit, $P(\text{"FALSE"})$ becomes one as anything is declared as "Unknown" or as another target. However, the $P(\text{TRUE} \& \text{"FALSE"})$ can only be as high as $P(\text{TRUE})$. The Negative Predictive Value, $P(\text{TRUE}|\text{"FALSE"})$, can be calculated as $P(\text{"FALSE"}\&\text{TRUE})/P(\text{"FALSE"})$. At the highest threshold, the numerator, $P(\text{"FALSE"}\&\text{TRUE})$, grows closer to $P(\text{TRUE})$ and the

denominator, $P(\text{"FALSE"})$ grows closer to one. Thus, an upper bound of $P(\text{TRUE})$ rather than one is found for Negative Predictive Values at the highest threshold.

Although the Vertical ROC curve is not a true ROC curve, it is still very useful. It can provide all the information about Vertical MOPs for a target at a glance. It also allows for easy comparison between algorithms. Finally, it can be used to show the impact uncertainty on prior populations of each target might have on an algorithm's effectiveness.

Calculating Vertical ROC Curves

Much of the calculations for Vertical ROC curves can be created from the information already gathered. Figure 15, showed again here as Figure 19, lists the information needed as well as showing how the information can be conveniently placed on each algorithm's worksheet for easy computation.

A	P(T)	0.2																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																	
---	------	-----	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--

$P(\text{"TRUE"}|\text{TRUE})$ is known, $P(\text{"FALSE"}|\text{TRUE})$ can be calculated by 1 minus this probability. Similarly, $P(\text{"FALSE"}|\text{FALSE})$ can be found. Thus, with the prior probabilities, $P(\text{"FALSE"} \& \text{TRUE})$ and $P(\text{"FALSE"} \& \text{FALSE})$ can be found. Their summation gives the $P(\text{"FALSE"})$. With $P(\text{"FALSE"})$ and $P(\text{"TRUE"})$ along with $P(\text{TRUE} \& \text{"TRUE"})$ and $P(\text{TRUE} \& \text{"FALSE"})$, both $P(\text{TRUE}|\text{"TRUE"})$ and $P(\text{TRUE}|\text{"FALSE"})$ can be found.

The compilation of Vertical ROC curve data is very similar to that of ROC curves. A table from the compilation sheet is given in Table 22, while the corresponding graph of the Vertical ROC curve is given in Figure 18.

Table 22. Vertical ROC Curve Table for Tiger.

Tiger	Threshold	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
TLS-Bound Curr	$P(T \sim T)$	0.047181										
	$P(T T)$	0.658417										
TLS-Bound	$P(T \sim T)$	0.047312	0.047312	0.047312	0.047312	0.047312	0.047145	0.050511	0.085868	0.14014	0.178205	0.2
	$P(T T)$	0.652968	0.652968	0.652968	0.652968	0.652968	0.659898	0.677183	0.69457	0.774856	0.836759	1
TLS & Delta Curr	$P(T \sim T)$	0.064426										
	$P(T T)$	0.669911										
TLS & Delta	$P(T \sim T)$	0.047312	0.047312	0.047312	0.047312	0.047312	0.073699	0.157111	0.193812	0.2	0.2	0.2
	$P(T T)$	0.652968	0.652968	0.652968	0.652968	0.652968	0.684591	0.872567	0.990972	1	1	1
Delta-Bound Curr	$P(T \sim T)$	0.110218										
	$P(T T)$	0.660884										
Delta-Bound	$P(T \sim T)$	0.066663	0.092868	0.110218	0.130265	0.148239	0.16394	0.178124	0.183779	0.193214	0.196925	0.2
	$P(T T)$	0.500059	0.602856	0.660884	0.759558	0.812019	0.923222	0.93453	0.972733	0.975757	0.982106	1

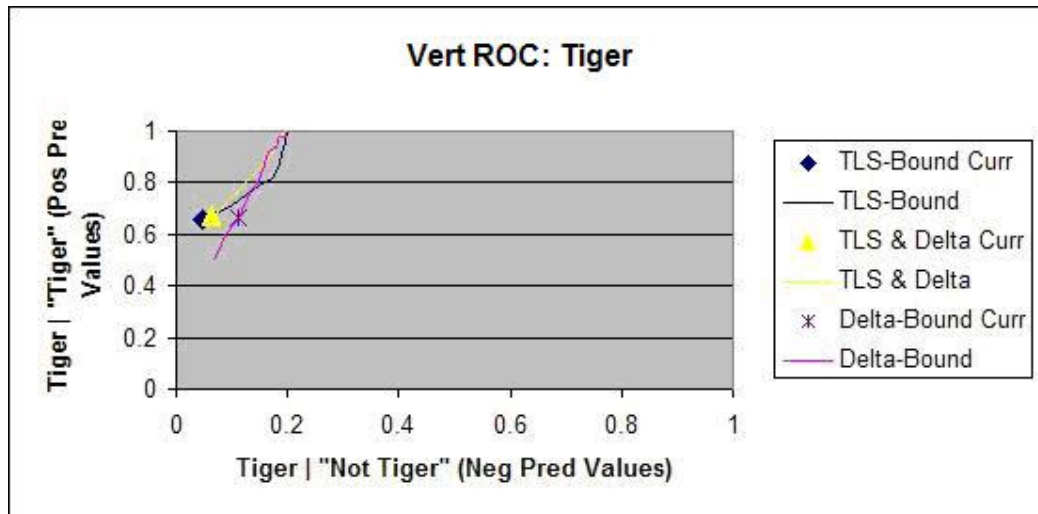


Figure 20. Vertical ROC Curve for Tiger.

The plot in Figure 20 suggests that if a better Positive Predictive Value is needed then the thresholds for all three of the algorithms are low. Table 22 suggests that a threshold around 0.6 or 0.7 for TLS-Bound or TLS & D would increase the Positive Predictive Value, but to the detriment of the Negative Predictive value. The threshold for “Tiger” is currently 0.49. The threshold for “T-72” is 0.605 and the Vertical ROC curve, given in Figure 21 looks much better.

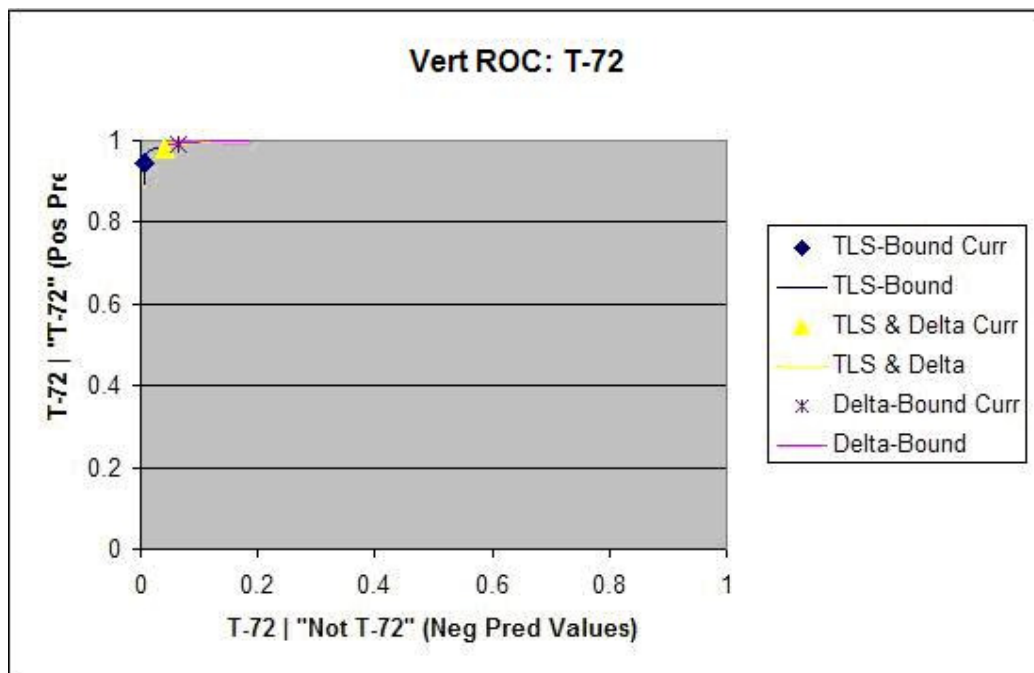


Figure 21. Vertical ROC Curve for T-72.

Figure 21 also shows a clearer difference between each of the three algorithms. TLS-Bound in this example has the best Negative Predictive Value, but the worst Positive Predictive Value. D-Bound is just the opposite, whereas TLS & D is in between. Also, the curves show how these points will be adjusted as the thresholds change. TLS-Bound can trade Positive Predictive Value for Negative Predictive Value simply by adjusting the threshold, whereas D-Bound can get worse in Negative Predictive Value with little to no improvement in Positive Predictive Value. The value

of the tradeoff is up to the decision-maker, but the information is present for a decision to be made.

As stated earlier, each of these Vertical ROC curves appear to have an upper limit for Negative Predictive Value. In the cases shown in Figures 20 and 21, the Negative Predictive Value, or $P(\text{TRUE} | \text{"FALSE"})$, seems bound at 0.2. The reason for this was stated before, but will be restated here. This is due to the prior probabilities of the example, which have the priors of all five targets at 0.2.

As the threshold increases, $P(\text{TRUE} | \text{"TRUE"})$ should increase as less False Positives are made. Also, $P(\text{"TRUE"})$ decreases to almost zero and $P(\text{"FALSE"})$ increases to one as it becomes more difficult to declare an event as "TRUE" with the higher threshold. The Negative Predictive Value can be calculated in terms of $P(\text{"FALSE"})$ and $P(\text{TRUE})$. As shown in the formulas below, the Negative Predictive value is bound by the Prior as the threshold increases.

$$\begin{aligned} P(\text{TRUE} | \text{"FALSE"}) &= P(\text{TRUE} \& \text{"FALSE"}) / P(\text{"FALSE"}) \\ &= P(\text{"FALSE"} | \text{TRUE}) * P(\text{TRUE}) / P(\text{"FALSE"}) \\ &= (\text{Value no greater than } 1) * \text{Prior} / 1 = \text{Prior} \end{aligned}$$

The higher the threshold, the higher the probability the event is TRUE when declared "FALSE," but only up to the probability of TRUE. Thus, as stated before, the Negative Predictive Values are bound by the Prior.

The bounds for Negative Predictive Values may decrease as new targets are considered, as each target will have a portion of the priors. To keep the Negative Predictive Value visually comparable, future analysis might attempt to scale the Vertical ROC curve graphs based on the priors. The UIT, however, will keep all graphs from zero to one to limit confusion in scales.

Prior Uncertainty

Another question with the Vertical ROC curve is how much the curves mean if the prior probabilities are uncertain. The means for entering priors into the UIT is shown earlier in Figure 10 and shown again here as Figure 22.

Target	T1	Tiger	Panzer	T-72	Bradley
Min Likeness	0.458	0.49	0.496	0.605	0.645
Min Delta	0.05	0.05	0.05	0.05	0.05
Highest Poss. Population	50	60	70	80	90
Most Likely Population	20	20	20	20	20
Lowest Poss. Population	10	10	10	10	10

Done Entering Thresholds

Figure 22. User-Defined Priors.

Three values for priors are asked for on each target. The values are the most likely population, the highest possible population and the lowest possible population. The data entered is not a probability, but an estimate of the number of units remaining. This will prevent probabilities greater than one to be mistakenly entered. The spawned worksheet will create a percentage for each target based on the populations entered. The values will be available on the “Algorithms” sheet for updates. This allows the decision-maker to enter the number of vehicles or targets likely to be encountered in an area and change it later as the operational environment changes.

The three values give many options for analysis. With the highest, lowest and most-likely values entered, a random sampling could be taken by forming a triangular distribution. Currently, the UIT instead calculates a highest possible percentage and lowest possible percentage in addition to the most likely percentage. Rather than sum the entire row to calculate these percentages as was done with the most likely population percentages, the UIT takes the highest possible population for a target and

adds the lowest possibly populations for all other targets to calculate the maximum possible percentage.

For example, Figure 22 shows the highest possible population for Tigers as 60. The lowest possible population for all other targets is each 10. The sum of these values is 100. Thus, the highest probability possible for encountering a Tiger occurs during the scenario when 60 Tigers exist as well as only 10 confusers, 10 Panzers, 10 T-72s and 10 Bradleys. The resulting probability is 60 divided by the sum of possible targets, 100. So, at most, the probability of encountering a Tiger is 60%. The minimum probability is calculated in a similar manner by taking the lowest possible population for a target and the highest possible populations from the other targets to calculate the lowest possible percentage per target.

Once the extremes to the priors are calculated, a Vertical ROC curve is created for each prior per target per algorithm. The resulting graph will be called a Priors curve. To generate these curves, worksheets for the three ROC-Friendly algorithms – TLS-Bound (A2), TLS & D (A3), and D-Bound (A5) – are created and named PriorsA2, PriorsA3 and Priors A5 respectively. Rather than copying all data, only the summary truth data is necessary. The truth data is not hard-copied over, but instead referenced by each of the sheets to the corresponding initial algorithm worksheet. The final worksheet looks like that shown in Figure 23.

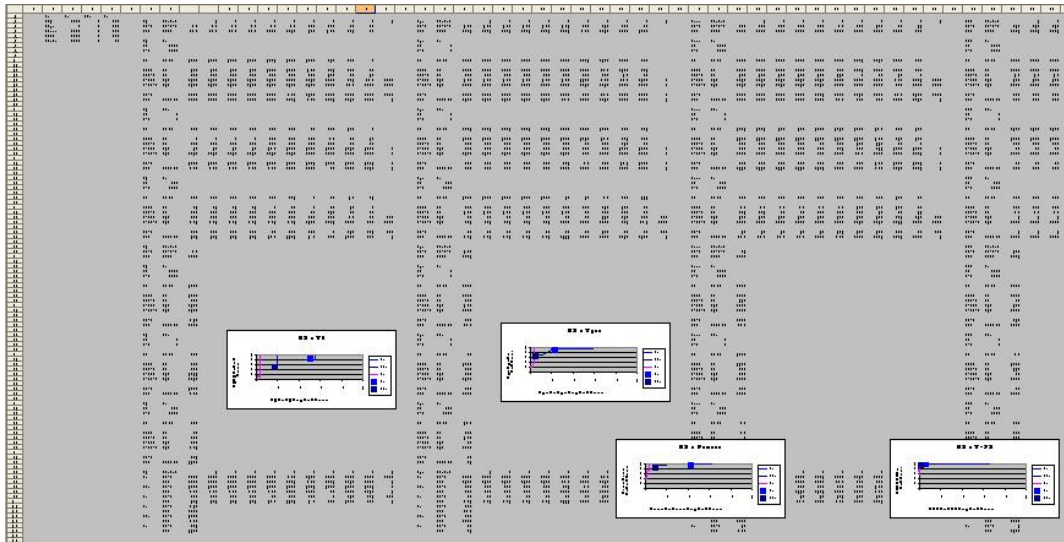


Figure 23. Snapshot of a Priors worksheet.

The values in the figure are too small to read, but it is provided just to give a general layout of the worksheet itself. The items of interest to the user would be the graphs running along the bottom of each of these worksheets. Two were moved up to show the data beneath. Two Priors curves are included as Figures 24 and 25.

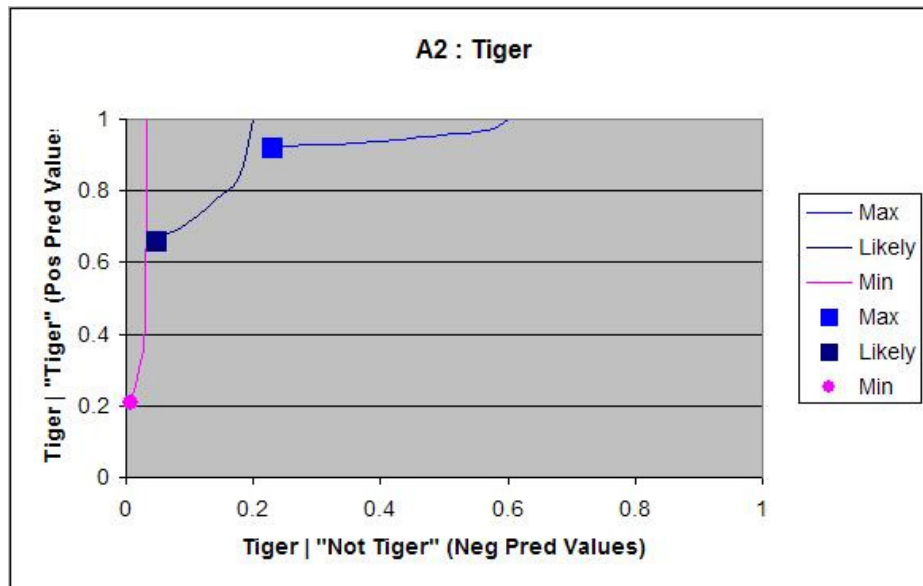


Figure 24. Priors Curve for Tiger.

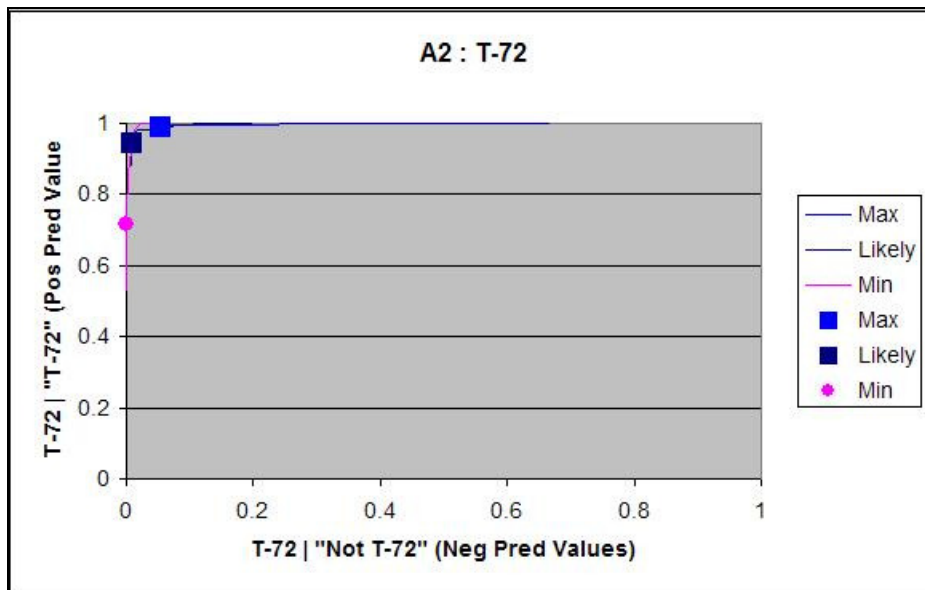


Figure 25. Priors Curve for T-72.

Both graphs show that the smaller the percentage of population, the lower both probabilities become. On the other hand, if the population is larger than expected, $P(\text{TRUE})$ is increased, so $P(\text{TRUE} | \text{"TRUE"})$ and $P(\text{TRUE} | \text{"FALSE"})$ tend to increase. How these values increase suggest ways of changing thresholds as priors change.

For example, the thresholds on "Tiger" should be increased if the population is suspected to be lower than stated, as the Positive Predictive Value climbs with little increase in Negative Predictive Value. However, thresholds should not be changed if the Tiger population might be higher than suspected. Alternatively, even if the population of T-72s is higher than suspected, the values do not change very much. Thus, the T-72 Vertical MOPs are not sensitive to more T-72s on the field. The thresholds for T-72s should still be raised if the population is lower than suspected.

Thus, the Priors curves show sensitivity to uncertainties in populations.

ROC Curves Based on Normalized Thresholds

Two final ROC curves can be made from this data. These are ROC curves to determine the validity of TLS and Delta as quantifiers. Since each In-Library target has both a TLS score and a Delta score, it is possible to normalize these scores and see how the targets compare to one another. By taking each target individually for each run, a target's score can be compared to the sum of the other scores. By making all negative Delta scores equal to 0, most scores will be bounded between 0 and 1. The only exception is where all Delta scores are zero. In this case, the run is excluded from analysis.

This ROC curve is more like a usual ROC curve than the other two, but it may be the least practical. The Normalized ROC curve will go from 0 to 1 in both True Positive and False Positive, since the scores are bound between 0 and 1, and the responses are limited to "Meets score" and "Does not meet score." If the target meets the score, then "TRUE" is considered declared. This will prove impractical, however, as each score considers only its value against the sum of the other values rather than the individual scores. For example, if the threshold for "Tiger" is set at 0.4, this computation will consider "Tiger" the response if it scores better than 0.4. However, a score of 0.4 means 0.6 was split amongst the other four targets. On the same run, the score for "Panzer" may be 0.5, in which case "Panzer" would've been selected, or the score for every other target may have been 0.15, making "Tiger" the only appropriate choice. Thus, the score in this case has no bearing on what any algorithm might actually return. The Normalized ROC curve will only help to judge the use of TLS and Delta as quantifiers.

To calculate scores for the first quantifier, TLS, the number of columns needed is one plus one for each In-Library target. The one individual column is the summation of all TLS-Scores. Next, Scores are calculated by dividing each target's TLS-Score by the summation of all TLS-Scores. Then, using a method similar to those applied to computing the original ROC curve data and the Vertical ROC curve data, summary truth information is calculated.

For the second quantifier, Delta scores, a few more columns are required. First, the original Delta scores should be copied, except with all negative values being considered as zero. Next, the summation of these values is calculated. Finally, scores are created by dividing the Delta score for each target by the summation as long as the summation is not zero. If the summation is zero, the data used to compute truth table will allow for a "U" to be returned in addition to the four coded letters – A, B, C and D – corresponding to different sections of the probability matrix. Any item with a "U" will be skipped by the COUNTIF() formulas.

In a similar manner with the ROC and Vertical ROC curves, the Normalized ROC curves are grouped by targets on one page. Rather than having three algorithms per graph, only the lines for the two quantifiers are combined on each graph. Two such graphs are provided as Figure 26 and 27.

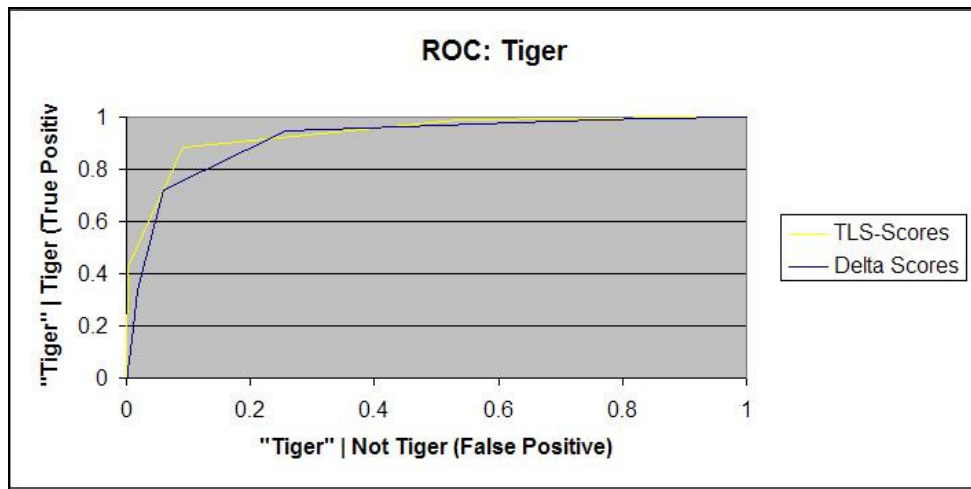


Figure 26. Normalized ROC Curve for Tiger.

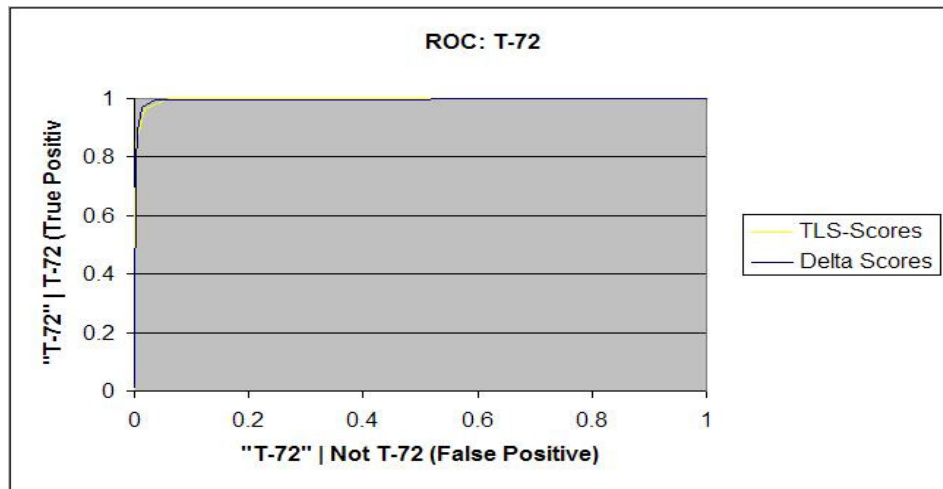


Figure 27. Normalized ROC Curve for T-72.

These figures show that both TLS and Delta provide insight into making the decision. Both quantifiers are much more useful in deciding whether a target is a T-72 or not as the Normalized ROC curve is in the far upper-left hand corner of Figure 27. This could possibly suggest a fusion technique that first considers whether a target is a T-72 or not may be appropriate. However, the graphs also show how only a few points identify the curves for TLS and Delta for the Tiger Normalized ROC curve.

This indicates consideration must be placed in which thresholds generate these curves.

Finally, none of these curves have a current operating condition, since none is possible.

Summary Statistics

Classifier Evaluation Quantifiers are described earlier as historical means to measure error. These quantifiers include Recall, Accuracy, F-Score, Precision and Mutual Information. These scores have been adjusted to work in a multiple declaration and error situation. The UIT uses one sheet to calculate these values and one sheet to display the result. A snapshot of the sheet used to calculate these values, entitled "Stats," is given in Figure 28.

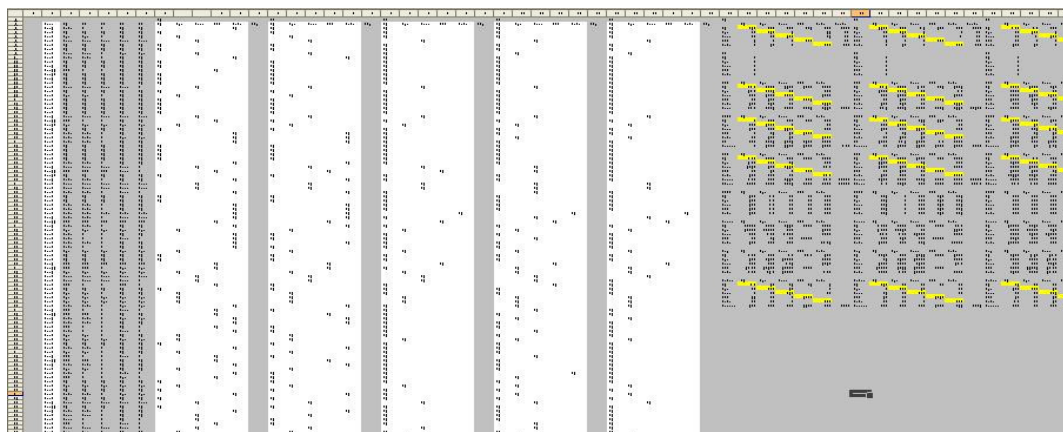


Figure 28. Snapshot of "Stats" worksheet.

The individual entries are too small to read, but the snapshot gives a layout of the worksheet. The worksheet starts with a list of all runs, the actual target being run against, and the response of each of the five algorithms. The five white bands represent each algorithm. In each white band, the In-Library targets are separated into columns. The name of the actual target is placed in the column of the In-Library target that was declared for that algorithm. These are counted for each five algorithms in the next section of the worksheet. The count for Biggest-TLS (A1) is shown in Table 23.

Table 23. Count for Biggest-TLS.

A1	T1	Tiger	Panzer	T-72	Bradley	Row Sum
T1	714	666	336	361	777	2854
Tiger	34	214	1	0	11	260
Panzer	23	33	153	0	51	260
T-72	1	4	2	253	0	260
Bradley	5	16	2	0	236	259

Table 23 shows actual targets down the left and In-Library targets as the heading of each column. Confusers have all been combined into T1 for this analysis. Each cell indicates the number of times the actual target is declared as the In-Library target. For example, Clutter (T1) is declared 714 times as “Clutter”, 336 times as “Panzer,” and 777 times as “Bradley.” The Row Sum indicates how many times the actual target was tested against.

By dividing all of the values in Table 23 by their Row Sum, $P(\text{“Declaration”} | \text{Actual})$ can be calculated. These values are shown in Table 24. The cells highlighted in yellow show where $P(\text{“TRUE”} | \text{TRUE})$ are found for each target. The sum of these values is called Recall and is shown in the bottom, right-hand corner. This value will be placed on the worksheet “ClassMetrics” with all other Classifier Evaluation Quantifiers for each algorithm.

Table 24. Recall Table.

$P(\text{“T”} \text{T})$	T1	Tiger	Panzer	T-72	Bradley	
T1	0.250175	0.233357	0.11773	0.126489	0.272249	
Tiger	0.130769	0.823077	0.003846	0	0.042308	
Panzer	0.088462	0.126923	0.588462	0	0.196154	
T-72	0.003846	0.015385	0.007692	0.973077	0	
Bradley	0.019305	0.061776	0.007722	0	0.911197	
Recall	0.250175	0.823077	0.588462	0.973077	0.911197	3.545987

The Priors are actually placed on the spreadsheet before the last table, but are not used until Accuracy needs to be calculated. By multiplying the values of the last graph by the priors, or multiplying $P(\text{“Declaration”} | \text{Actual}) * P(\text{Actual})$, the $P(\text{Actual} \& \text{“Declaration”})$ can be found. The result is given in Table 25. Accuracy is defined as $P(\text{TRUE} \& \text{“TRUE”})$ plus $P(\text{FALSE} \& \text{“FALSE”})$ (Wallach, 2004). For multiple

declarations, it shall be redefined here as $P(\text{TRUE} \& \text{"TRUE"})$ for each target. These values will be highlighted in the worksheet, as they are highlighted here, in yellow. The resulting Accuracy is in the bottom, right-hand corner. Table 25 also shows that before the Accuracy is calculated, the probabilities of declarations, $P(\text{"Declaration"})$ for each target are also calculated.

Table 25. Accuracy Table.

P(T&"T")	T1	Tiger	Panzer	T-72	Bradley	
T1	0.050035	0.046671	0.023546	0.025298	0.05445	
Tiger	0.026154	0.164615	0.000769	0	0.008462	
Panzer	0.017692	0.025385	0.117692	0	0.039231	
T-72	0.000769	0.003077	0.001538	0.194615	0	
Bradley	0.003861	0.012355	0.001544	0	0.182239	
P("T")	0.098511	0.252103	0.14509	0.219913	0.284382	
Accuracy	0.050035	0.164615	0.117692	0.194615	0.182239	0.709197

Precision is $P(\text{TRUE} | \text{"TRUE"})$ and F-Score is the harmonic mean between Precision and Recall (Wallach, 2004). By dividing each cell in Table 25, $P(\text{Actual} \& \text{"Declaration"})$ by $P(\text{"Declaration"})$, the $P(\text{Actual} | \text{"Declaration"})$ can be obtained. The possibility of dividing against zero is guarded against. These values are shown in Table 26. $P(\text{TRUE} | \text{"TRUE"})$ is the definition of precision (Wallach, 2004). With multiple declarations, it is redefined here again as the sum of these values for each target. These values are highlighted in yellow below. In addition, an F-Score is evaluated using the Precision and Recall of each target and the sum of the F-Scores is also calculated in Table 26.

Table 26. Precision and F-Score Table.

P(T T')	T1	Tiger	Panzer	T-72	Bradley	
T1	0.507911	0.473766	0.239017	0.256801	0.552727	
Tiger	0.103743	0.652968	0.003051	0	0.033564	
Panzer	0.12194	0.174957	0.811166	0	0.270389	
T-72	0.003498	0.013992	0.006996	0.884964	0	
Bradley	0.013577	0.043446	0.005431	0	0.640827	
Precision	0.507911	0.652968	0.811166	0.884964	0.640827	3.497836
F Score	0.33523	0.72822	0.682096	0.926931	0.752462	3.424939

The final Classifier Evaluation Quantifier suggested is Mutual Information (Wallach, 2004). This value involves numerous steps include a logarithm, quotient

and two products. These steps are each accomplished by the UIT and the results are shown in Table 27.

Table 27. Mutual Information Table.

P(T)*P("T")	T1	Tiger	Panzer	T-72	Bradley	
T1	0.019702	0.050421	0.029018	0.043983	0.056876	
Tiger	0.019702	0.050421	0.029018	0.043983	0.056876	
Panzer	0.019702	0.050421	0.029018	0.043983	0.056876	
T-72	0.019702	0.050421	0.029018	0.043983	0.056876	
Bradley	0.019702	0.050421	0.029018	0.043983	0.056876	
Quotient	T1	Tiger	Panzer	T-72	Bradley	
T1	2.539555	0.925639	0.811422	0.575178	0.957339	
Tiger	1.327452	3.264838	0.026509	0	0.148771	
Panzer	0.897983	0.503456	4.05583	0	0.689756	
T-72	0.039043	0.061025	0.053017	4.424822	0	
Bradley	0.195967	0.245042	0.053222	0	3.204135	
Log	T1	Tiger	Panzer	T-72	Bradley	
T1	0.404758	-0.03356	-0.09075	-0.2402	-0.01893	
Tiger	0.123019	0.513862	-1.57661	0	-0.82748	
Panzer	-0.04673	-0.29804	0.60808	0	-0.1613	
T-72	-1.40846	-1.21449	-1.27558	0.645896	0	
Bradley	-0.70782	-0.61076	-1.27391	0	0.505711	
Product	T1	Tiger	Panzer	T-72	Bradley	
T1	0.020252	-0.00157	-0.00214	-0.00608	-0.00103	
Tiger	0.003217	0.08459	-0.00121	0	-0.007	
Panzer	-0.00083	-0.00757	0.071566	0	-0.00633	
T-72	-0.00108	-0.00374	-0.00196	0.125701	0	
Bradley	-0.00273	-0.00755	-0.00197	0	0.09216	
Mutual Info	0.018826	0.064175	0.064287	0.119625	0.0778	0.344712

Once all the Classifier Evaluation Quantifiers are calculated, they are presented in the worksheet, "ClassMetrics," as Table 28 shows.

Table 28. Classifier Evaluation Quantifiers.

	A	B	C	D	E	F	G
1							
2		Stats	A1	A2	A3	A4	A5
3		Recall	3.545987	3.715924	3.027143	2.944967	3.032921
4		Accuracy	0.709197	0.743185	0.605429	0.588993	0.606584
5		Precision	3.497836	3.733636	3.800473	3.453918	3.544572
6		FScore	3.424939	3.660544	3.04015	2.898849	2.978159
7		M Info	0.344712	0.369084	0.283576	0.245011	0.253836

Another quantifier was originally suggested called a Cost Matrix, but proved too unreliable to include at this stage of the UIT. The Cost Matrix uses designations of "Neutral," "Friend," "Target of the Day," "Other Hostile," "Out-of-Library," and "Non-Declaration" for each In-Library target and each actual target. A prior probability was

necessary for each Actual target. The Cost Matrix was then determined by calculating conditional probabilities, multiplying them by the suggested cost matrix, and then adding all the costs for each condition. The three steps – conditional probability, cost matrix, individual costs – are shown in Figure 29.

Cost	Declar's	T1	T2	T3	T4	T5	U	Non-Dec
Given	Type	"N"	"F"	"F"	"TOD"	"OH"	"OOL"	"Non-dec"
T1	N							
T2	F	0.0082	0.0550	0.0003	0.0000	0.0021	0.0013	
T3	F	0.0053	0.0080	0.0368	0.0000	0.0000	0.0013	
T4	TOD	0.0000	0.0000	0.0000	0.0550	0.0000	0.0005	
T5	OH	0.0013	0.0041	0.0000	0.0000	0.0606	0.0000	
Coa1	OOL	0.0143	0.0170	0.0031	0.0000	0.0277	0.0041	
Coa2	OOL	0.0030	0.0103	0.0067	0.0021	0.0041	0.0347	
Coa3	OOL	0.0041	0.0062	0.0032	0.0023	0.0015	0.0423	
Coa4	OOL	0.0170	0.0133	0.0082	0.0015	0.0085	0.0167	
Coa5	OOL	0.0357	0.0123	0.0018	0.0000	0.0033	0.0126	
Coa6	OOL	0.0267	0.0126	0.0046	0.0015	0.0121	0.0030	
Coa7	OOL	0.0175	0.0244	0.0031	0.0003	0.0198	0.0018	
Coa8	OOL	0.0175	0.0195	0.0031	0.0000	0.0254	0.0013	
Coa9	OOL	0.0177	0.0200	0.0021	0.0000	0.0262	0.0008	
Coa10	OOL	0.0072	0.0162	0.0203	0.0023	0.0118	0.0030	
Coa11	OOL	0.0053	0.0077	0.0136	0.0311	0.0013	0.0072	
Coa12	OOL	0.0005	0.0003	0.0000	0.0000	0.0000	0.0000	

Cost	Declar's	T1	T2	T3	T4	T5	U	Non-Dec
Target	Type	"N"	"F"	"F"	"TOD"	"OH"	"OOL"	"Non-dec"
T1	N	-20	-20	-20	100	100	0	5
T2	F	-20	-20	-20	100	100	0	5
T3	F	-20	-20	-20	100	100	0	5
T4	TOD	100	100	100	-20	10	10	5
T5	OH	100	100	100	-20	10	10	5
Coa1	OOL	0	0	0	10	10	-20	5
Coa2	OOL	0	0	0	10	10	-20	5
Coa3	OOL	0	0	0	10	10	-20	5
Coa4	OOL	0	0	0	10	10	-20	5
Coa5	OOL	0	0	0	10	10	-20	5
Coa6	OOL	0	0	0	10	10	-20	5
Coa7	OOL	0	0	0	10	10	-20	5
Coa8	OOL	0	0	0	10	10	-20	5
Coa9	OOL	0	0	0	10	10	-20	5
Coa10	OOL	0	0	0	10	10	-20	5
Coa11	OOL	0	0	0	10	10	-20	5
Coa12	OOL	0	0	0	10	10	-20	5

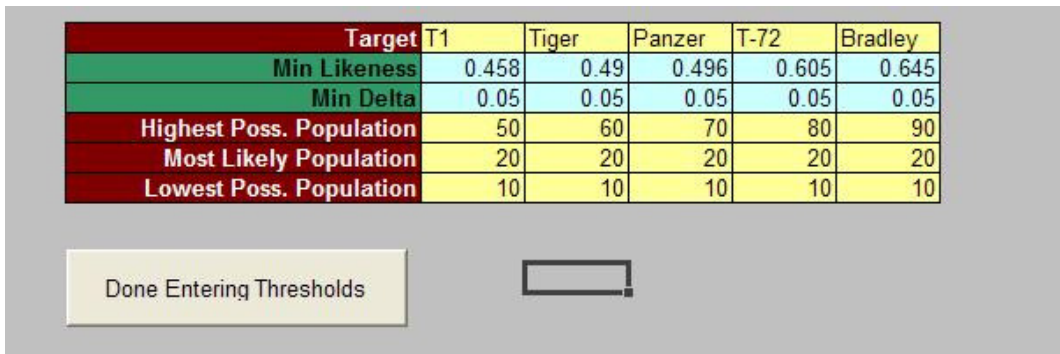
Cost	Declar's	T1	T2	T3	T4	T5	U	Non-Dec
Target	Type	"N"	"F"	"F"	"TOD"	"OH"	"OOL"	"Non-dec"
T1	N	0	0	0	0	0	0	0
T2	F	-0.164338	-1.033403	-0.005137	0	0.205431	0	0
T3	F	-0.118161	-0.15326	-0.775751	0	1.284337	0	0
T4	TOD	0.025687	0.051374	0.051374	-1.233763	0	0.005137	0
T5	OH	0.123426	0.410934	0.081274	0	-1.212433	0	0
Coa1	OOL	0	0	0	0	0.277421	-0.082199	0
Coa2	OOL	0	0	0	0.02055	0.041033	-0.633552	0
Coa3	OOL	0	0	0	0.028256	0.015412	-0.85735	0
Coa4	OOL	0	0	0	0.015412	0.084768	-0.333333	0
Coa5	OOL	0	0	0	0	0.033333	-0.251734	0
Coa6	OOL	0	0	0	0.015412	0.12073	-0.17381	0
Coa7	OOL	0	0	0	0.002563	0.137731	-0.035362	0
Coa8	OOL	0	0	0	0	0.254303	-0.025687	0
Coa9	OOL	0	0	0	0	0.262003	-0.015412	0
Coa10	OOL	0	0	0	0.023115	0.118161	-0.17381	0
Coa11	OOL	0	0	0	0.310814	0.012644	-0.143848	0
Coa12	OOL	0	0	0	0	0	0	0

Figure 29. Steps for computing a Cost Matrix.

The Cost Matrix is not included in the UIT at this stage for multiple reasons. It relies on correct priors that are estimates. It relies on costs that are very subjective. It requires designations which were not provided. It also requires the probabilities of all the confusers. With so many unknowns, this quantifier is explored and documented, but not added to the UIT at this time.

IV. Results and Analysis

The results from a possible analysis on the sample data is outlined in this section to show how a study using the tool could proceed. The results are determined using the same 3893 experimentation runs provided in the sample data. None of the user-defined parameters – TLS-threshold, Delta-threshold, nor priors – were given at the beginning of the study, as parameters can be estimated by the user, the decision maker, or the subject matter expert. The sample parameters used for the first round of analysis are given in Figure 30.



Target	T1	Tiger	Panzer	T-72	Bradley
Min Likeness	0.458	0.49	0.496	0.605	0.645
Min Delta	0.05	0.05	0.05	0.05	0.05
Highest Poss. Population	50	60	70	80	90
Most Likely Population	20	20	20	20	20
Lowest Poss. Population	10	10	10	10	10

Done Entering Thresholds

Figure 30. Sample data parameters.

The values for “Most Likely Population” are chosen to give all targets equal priors. Thus, all targets are given the equivalent number of 20 units as a most likely population. Likewise, the lowest possible populations for each target were set to equal values of 10 for some stability in calculations. For some contrast, the values for “Highest Population” vary slightly, from 50 to 90 in increments of 10. The effects of these values will be studied during the sensitivity analysis of prior probabilities portion of the study.

The TLS-threshold values, identified as “Min Likeness,” are specifically chosen in this sample analysis by first inspecting the data to look for the lowest TLS Score each

actual target will return when compared to itself. During the inspection, the data was separated into groups of actual targets. Only the experimental runs against that actual target are considered in each graph. Next, for each of these experimental runs the TLS comparing the actual target to its In-Library target equivalent is taken. From these values, the lowest possible True Score, TLS for the target when compared to its In-Library equivalent, is selected as the TLS-threshold for that target.

For example, when Tigers were tested as the actual target, the identification system returned TLS values ranging from 0.49 to 1.00 when comparing the actual Tiger to its In-Library “Tiger.” This range of True Scores can be seen in the scatterplot of Tiger (T2) scores shown earlier and provided again here as Figure 31. The scatterplot also shows the scores for the other targets at each True Score encountered in the experiment.

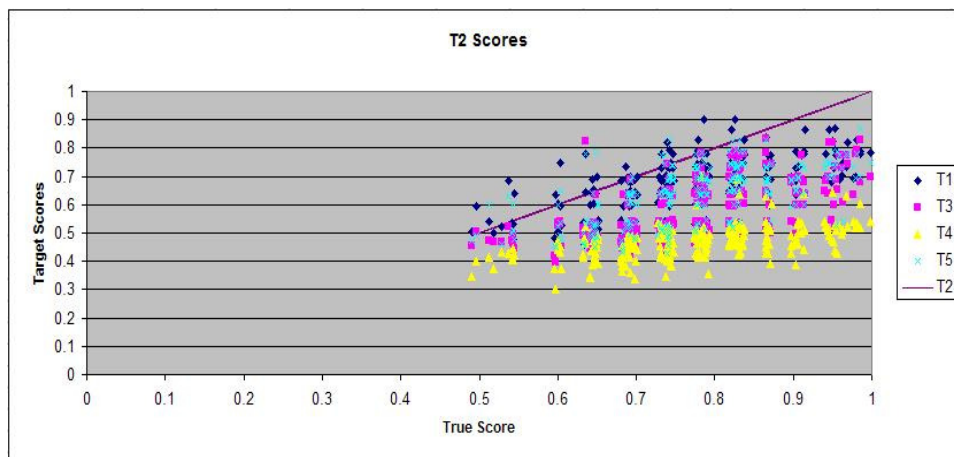


Figure 31. Tiger scores showing True Scores range between 0.49 and 1.0

Thus, the lowest return for “TRUE” when the signal is TRUE for Tigers is a 0.49 for this sample. Any return less than this value is clearly FALSE. This value is chosen for the baseline TLS-threshold so only FALSE identifications are eliminated by

this threshold. The actual TLS-threshold should be no less than this value unless an analysis on Delta scores is accomplished. In contrast, Delta-thresholds are set at 0.05 to eliminate some positives to allow for some comparison between the algorithms.

Thus, all user-defined parameters are clearly identified for the first analysis. Since the spreadsheet is dynamic, the parameters may be adjusted later during further rounds of analysis. A good use for the first study is to determine how the parameters should change. Although the best algorithm is sought, each algorithm is partially defined by the parameters and well-chosen parameters can improve each algorithm. Once each algorithm is at its best, the best of the algorithms can be chosen.

First Analysis

After all parameters are entered, the analysis begins by taking a look at ROC curves. As stated previously, ROC curves will only show for the algorithms TLS-Bound, TLS & D, and D-Bound. The definitions for these algorithms shall be repeated here as a reminder of what they involve.

- A1 or Biggest-TLS: (Forced) The target with the highest TLS is returned.
- A2 or TLS-Bound: (Not Forced) The target with the highest TLS is returned if it meets the TLS-Threshold. Otherwise, “Unknown” is returned.
- A3 or TLS & D: (Not Forced) The target with the highest TLS is returned if it has the largest Delta and meets the TLS-Threshold. Otherwise, “Unknown” is returned.
- A4 or Big D: (Forced) The target with the highest Delta score is returned.

- A5 or D-Bound: (Not Forced) The target with the highest Delta is returned if it meets the minimum Delta-Threshold. Otherwise, “Unknown” is returned.

Again, Biggest-TLS and Big-D do not have thresholds to adjust, so they are not plotted separately. However, Biggest-TLS is a special case of TLS-Bound where the TLS-threshold is zero, so its result can be found at the end of the TLS-Bound line. Similarly, Big-D can be found at the end of the D-Bound line.

The ROC curve for Clutter (T1) is not provided, since increasing the threshold would, as minimum thresholds are no longer met, change “Clutter” declarations to “Unknown” declarations which are still appropriate for Clutter. The ROC curve for every other In-Library target is given as Figures 32 to Figure 35.

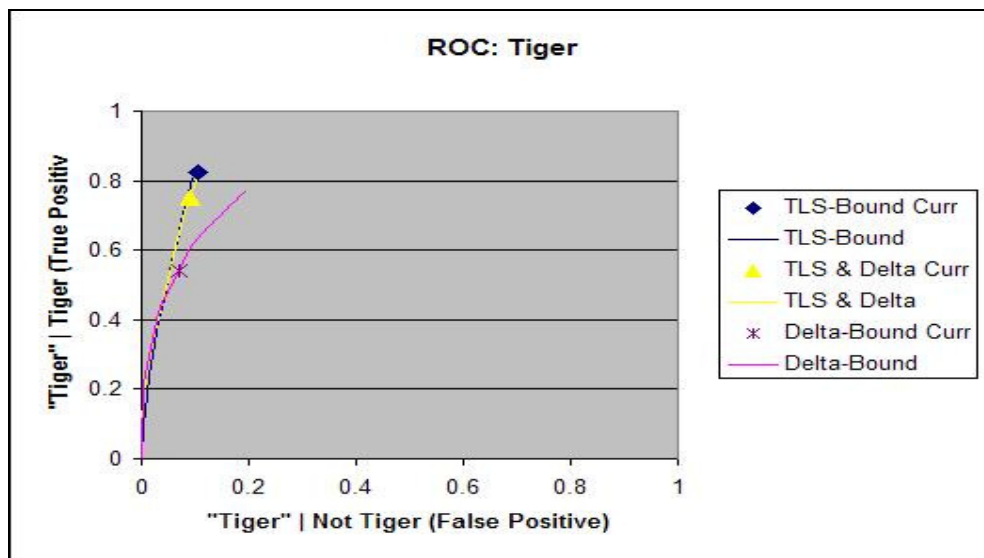


Figure 32. ROC Curve for Target 2, Tiger.

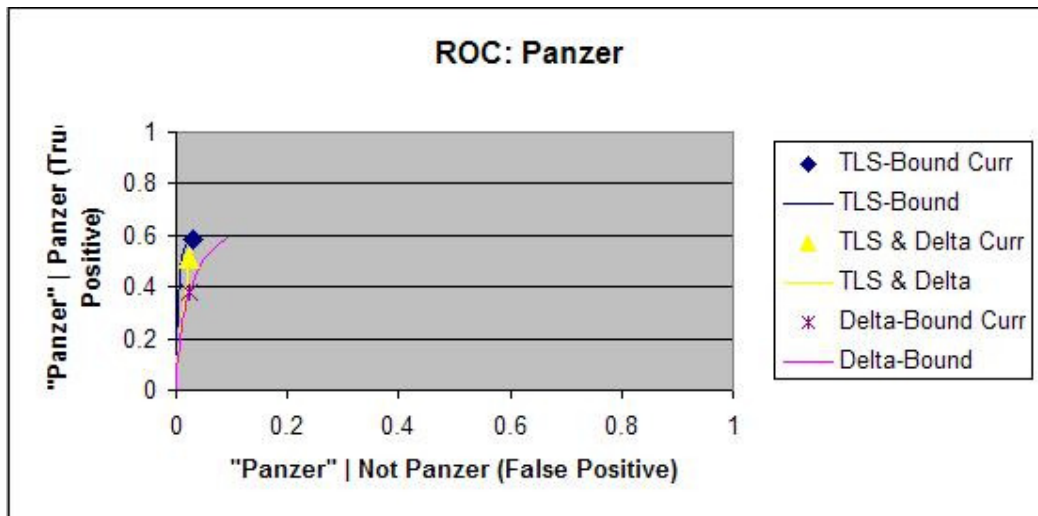


Figure 33. ROC Curve for Target 3, Panzer.

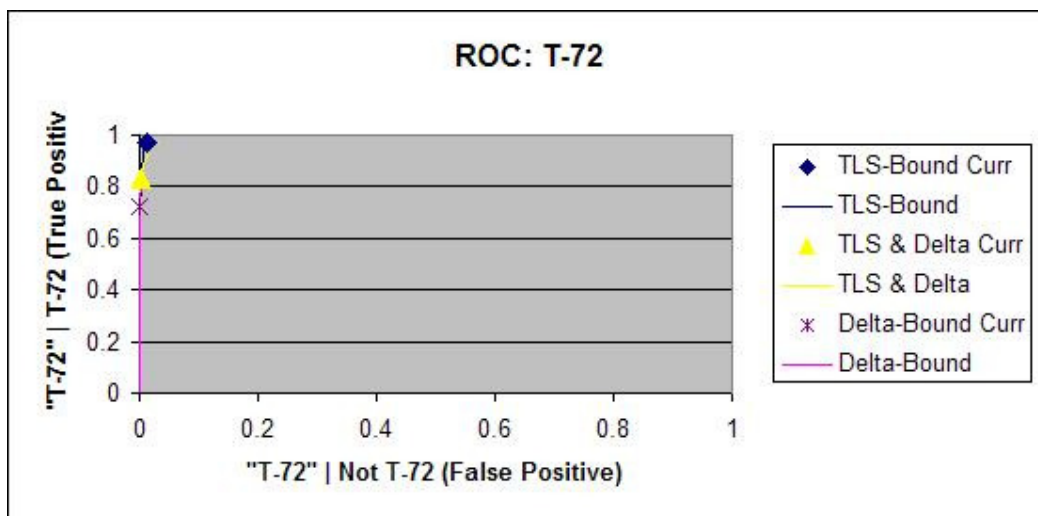


Figure 34. ROC Curve Target 4, T-72.

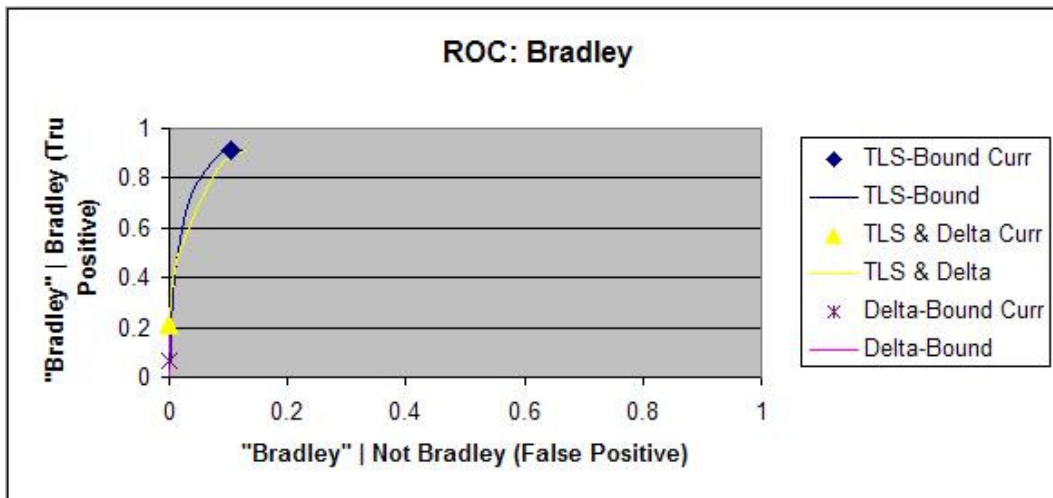


Figure 35. ROC Curve Target 5, Bradley.

The current operating point, indicated by a dot for each algorithm, for the TLS-Bound algorithm is always near the top True Positive part of the ROC curve for its algorithm. This is due to the use of the minimum True Score for each target as their individual thresholds. The current operating point may not be at the very top, because instances where the True Score is equal to the highest TLS value are still returned as “Unknown.” Adjusting the TLS-threshold above the minimum True Score would move the current operating point along the ROC curve. However, it could change algorithms based on Delta as discussed in Section III, Methodology. Examples of this are shown in Figure 36.

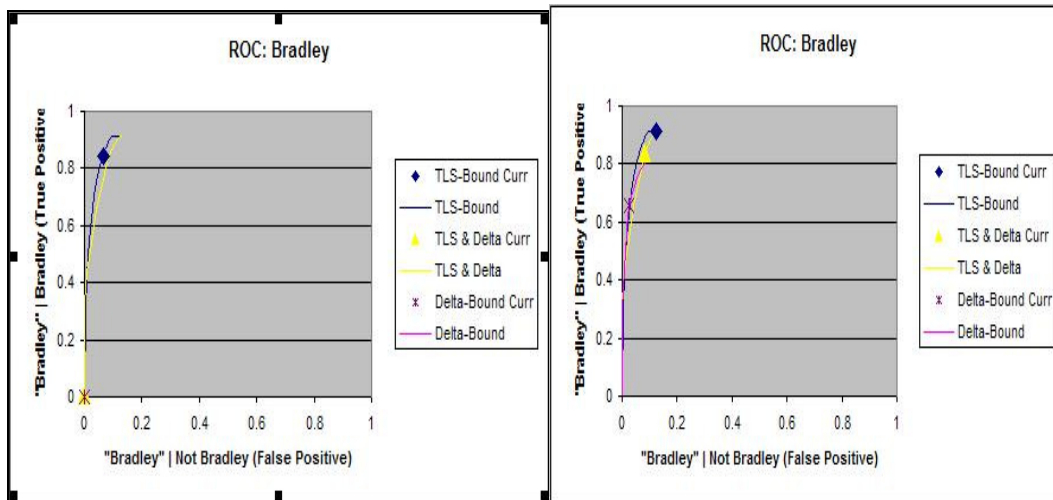


Figure 36. Adjusted Thresholds: Left (High Threshold), Right (Low).

Figure 36 shows that by increasing the TLS-threshold for Bradley from 0.645 to 0.7, the current operating point for TLS-Bound moved down the ROC curve. In addition, the points or curves for the Delta-based algorithms reduced as the Deltas are affected. On the other hand, lowering the TLS-threshold increased the curve for D-Bound, suggesting the threshold was too high to allow for competitive Delta scores. Adjusting Delta-thresholds will further move the current operating point for Delta-based algorithms along their ROC curves.

For TLS-Bound, the current operating point will only move down the ROC curve as correct identifications become “Unknown” declarations due to an increased threshold. If an increased threshold causes one target that was declared as a different target, called a misidentification, to instead become a declaration of “Unknown,” the ROC curve will not change as “Unknown” declarations count as misidentifications do: as “FALSE.” At maximum threshold, only “Unknown” declarations are made, and the probability of a true positive is zero.

Alternatively, the False Positive rates seem to reach an upper limit around 0.2 . With five targets, identifications are spread throughout each of the five targets. Thus, as the thresholds are set at their worst conditions, it is expected that one out of five, or 0.2, is about average for how bad a system can do. As the threshold increases, declarations no longer occur and the probability of a False Positive reduces to zero. Also of note, the current operating condition can be a point off of the curve, since the curve is a smoothed line joining eleven points for thresholds along each algorithm. More detail on how ROC curves are generated is included in Section III, Methodology.

The ROC curve indicates that T-72 far surpasses the other targets in performance as its ROC curve is the closest to the upper left. The Bradley also shows a good ROC curve for the two algorithms based on TLS, but the current operating point for TLS & D is bad. As these two targets have high TLS-thresholds, this could indicate a higher TLS-threshold being more appropriate.

All of the ROC curves show the TLS-Bound as the best algorithm using these quantifiers. This is indicated as the blue curve and current operating point are higher than the others, showing better true positive values. However, the current operating point is generally further right than the current operating points for the other two algorithms. This indicates that under current conditions, the TLS-Bound algorithm will provide the best True Positive probability in return for a less favorable False Positive probability. Determining which algorithm is the best requires the preference of these two probabilities to be weighed.

The Bradley ROC curve for TLS & D shows the current operating point is a low point on the TLS & D curve. This means either the TLS-threshold for the Bradley is too high, or the other TLS-thresholds are too low. By eliminating only the

obviously wrong events in the TLS-Bound algorithm, the high threshold prevented the algorithm from having a Delta that could beat the Delta of the other algorithms. Thus, the TLS-threshold for Bradleys should be lowered or the TLS thresholds for the other targets increased. The current operating point for the Tiger using the D-Bound (All) algorithm lies in the middle of the curve. This shows the Delta threshold for Tiger is too high. A lower threshold would only affect Tiger, and significantly raise True Positives with a moderate raise in False Positives. To show how this would work, the Delta-threshold was changed from 0.05 to -1.00 and the results are shown below in Figure 34.

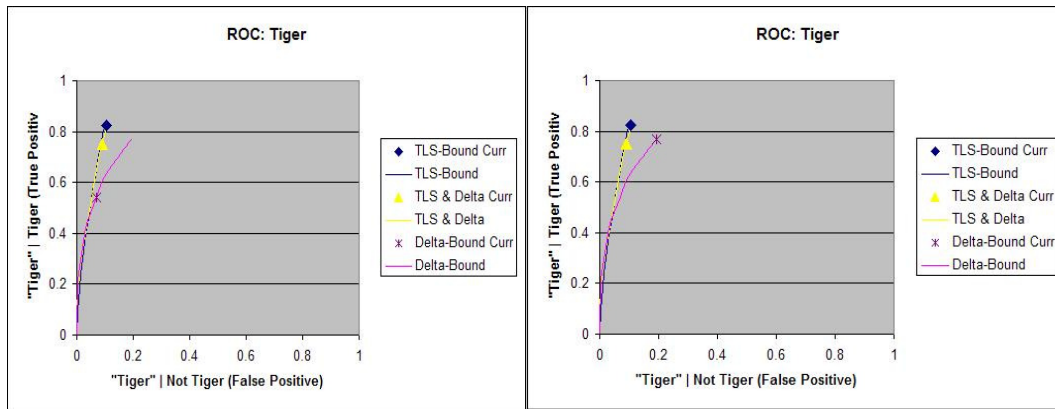


Figure 37. D-Bound Current point changing with Delta-threshold.

The ROC curves provided many insights into the system. They showed which TLS-thresholds could be increased or reduced and which Delta-thresholds could be increased and reduced. They also show TLS-Bound clearly outperforming the other two algorithms in True Positive probabilities. However, they only provide information at the system level.

A modified ROC curve, the Vertical ROC curve, is needed to show how parameters and algorithms would perform from the user's point of view. These curves are given as Figure 38 to Figure 41.

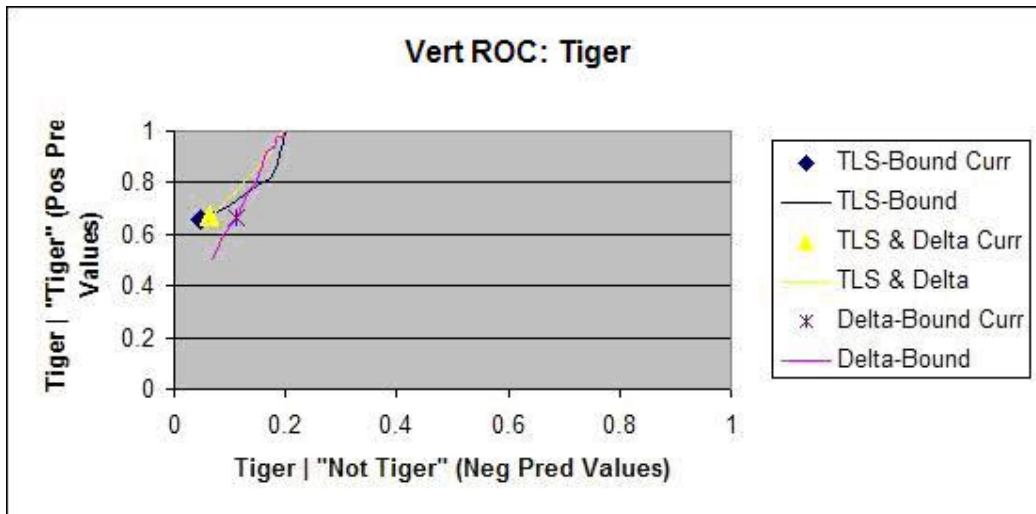


Figure 38. Vertical ROC Curve for Target 2, Tiger.

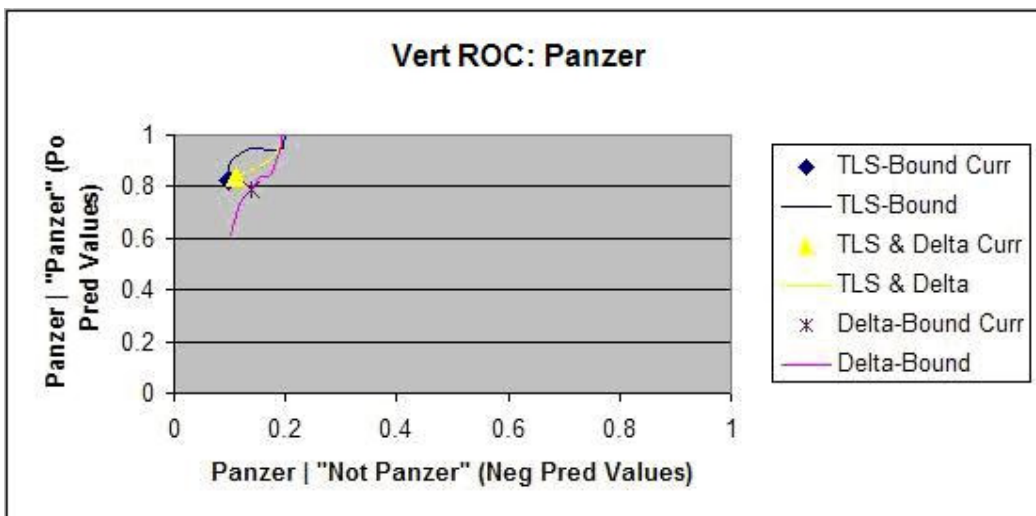


Figure 39. Vertical ROC Curve for Target 3, Panzer.

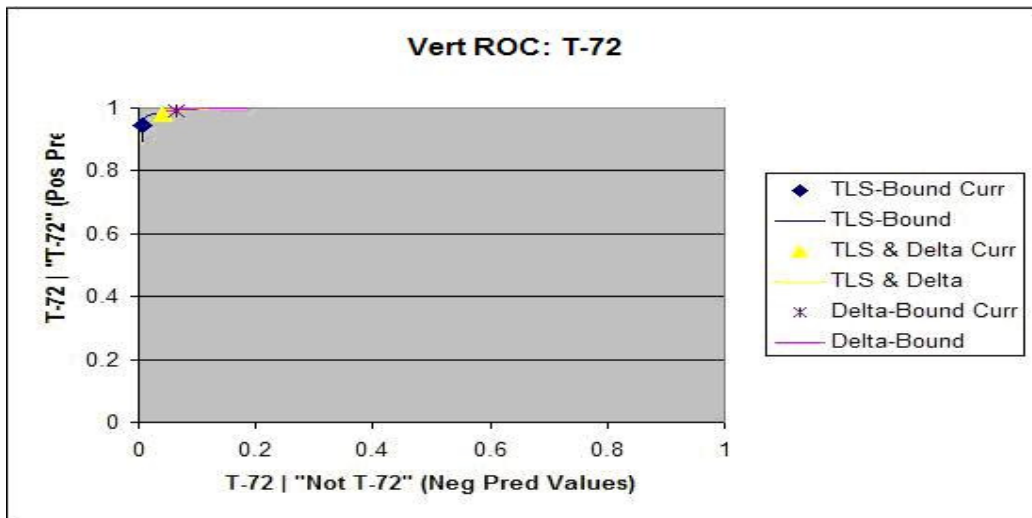


Figure 40. Vertical ROC Curve for Target 4, T-72.

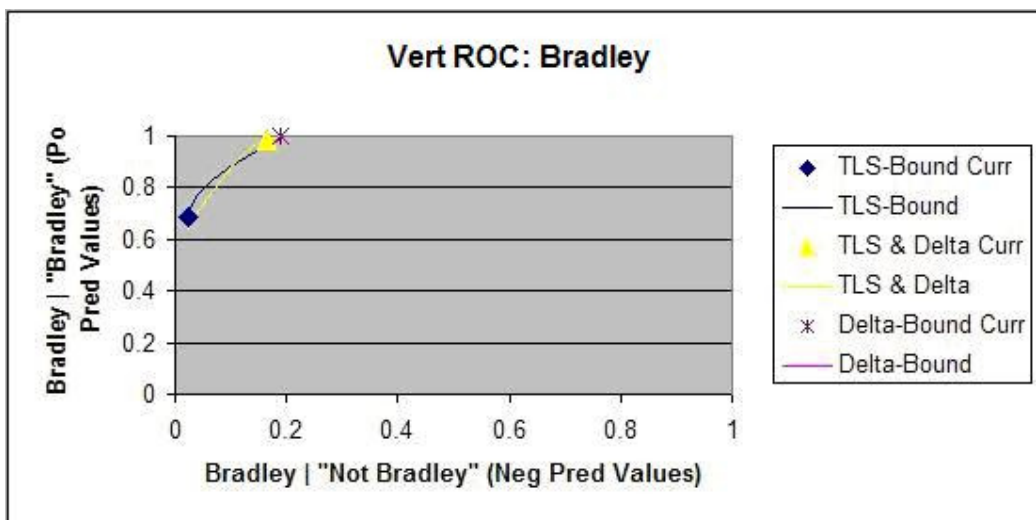


Figure 41. Vertical ROC Curve for Target 5, Bradley.

Unlike the graphs for the ROC curves, these show the current operating conditions for TLS-Bound to be on the low end of the curve. This is because a lower threshold results in more declarations and more of a chance that other targets were declared “TRUE.” The False Negatives seem to have an upper bound at 0.2 . This occurrence is explained in Section III, Methodology, but shall be repeated briefly here.

At the highest threshold, the probability of a “FALSE” or “Other” declaration goes to one as everything is declared as “Unknown.” The probability of it being TRUE can only be as high as the prior probability, which was entered earlier as 0.2 for all targets. Thus, the probability of TRUE and “FALSE” is 0.2, since all declarations are “Unknown” or “FALSE.” Thus the Negative Predictive Value, $P(\text{TRUE} | \text{“FALSE”})$, equals the $P(\text{TRUE} \& \text{“FALSE”})$ divided by $P(\text{“FALSE”})$ which is 0.2 times 1 divided by 1. The resulting answer is 0.2. Along with a more in-depth explanation of this bound, further calculations for generating the Vertical ROC curve are provided in Section III, Methodology.

Again, the T-72 identification outshines the other identifications as the Vertical ROC curve for T-72 remains in the upper left hand corner. Using TLS-Bound, the rest show Positive Predictive Values from 0.65 to 0.8 with Negative Predictive Values ranging from very small to 0.1. This means, the probability the declaration is correct will be around 0.7, but the probability it was detected as something else or called an “Unknown” is less than 0.1.

It should be reminded that “Unknown” is considered in the “Not a Tiger,” “Not a Panzer,” “Not a T-72,” and “Not a Bradley” categories as it is counted as “Clutter.” Thus, non-detections and unknowns are shown in the graphs as misidentifications. As the threshold increases, more “Unknowns” are produced and the graphs reflect it with the increased Negative Predictive Value.

When the Vertical ROC curve line is horizontal, increasing the threshold will cause a large unfavorable increase in Negative Predictive Value for a smaller return in improving the Positive Predictive Value. However, the line for Panzer is initially vertical, indicating a good tradeoff will occur if the threshold is increased. Thus,

increasing the TLS-threshold will ensure a “Panzer” is actually a Panzer without causing a significant increase in misidentifications actually being Panzers. The Bradley Vertical ROC curve with a diagonal line as a slope shows an equivalent tradeoff. It is up to the user as to which, the Positive Predictive Value or the Negative Predictive Value, is worth more and consequently what balance between Positive and Negative Predictive Value is favorable.

Thus the Vertical ROC curves answer many questions about the Vertical MOPs. The Panzer Vertical MOPs can be improved by increasing the TLS-threshold. TLS-Bound or Biggest-TLS are the best ways to go, especially if low Negative Predictive Values are desired. The T-72 identification is working wonderfully. The Bradley and Tiger algorithms can be adjusted, but any improvement to Positive Predictive Value will hurt the Negative Predictive Values.

A quick look at the Normalized ROC curve will show how much information is being given by TLS and by Delta scores. TLS is again the original score, or Target Likeness Score, relating how like an actual target is to an In-Library target. The Delta score is the difference between a TLS and the TLS-threshold for the target. The TLS-thresholds are user-defined parameters set for each target individually. A good ROC curve, with values closer to the upper left, will indicate the quantifier is providing much useful information. The Normalized ROC curves for all four targets are provided in Figures 42 to Figure 45.

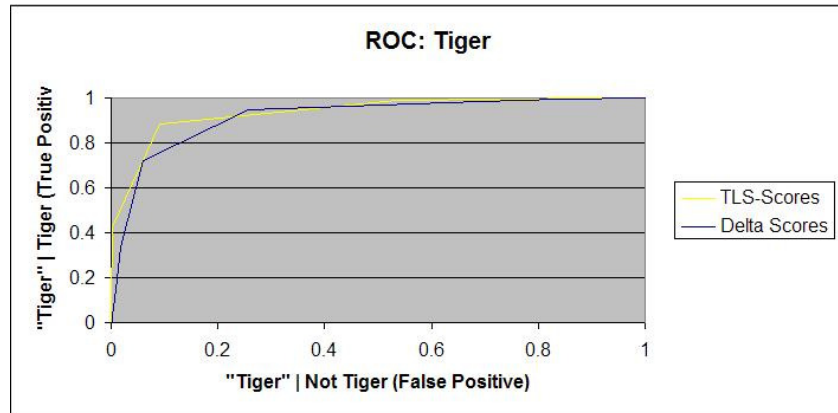


Figure 42. Normalized ROC Curve for Target 2, Tiger.

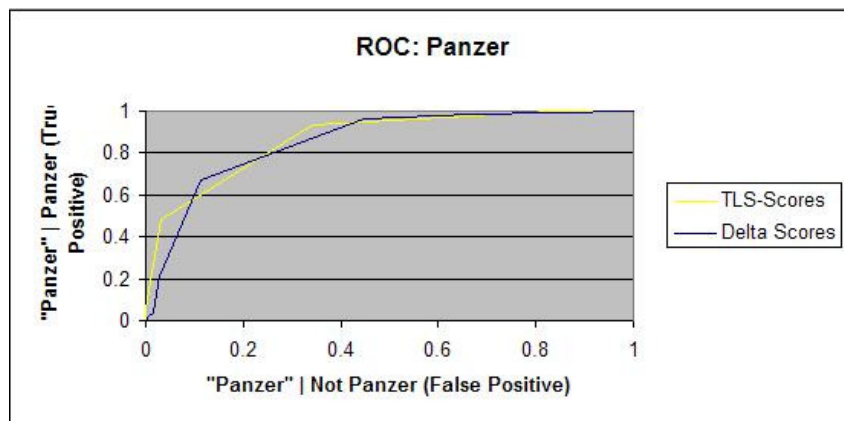


Figure 43. Normalized ROC Curve for Target 3, Panzer.

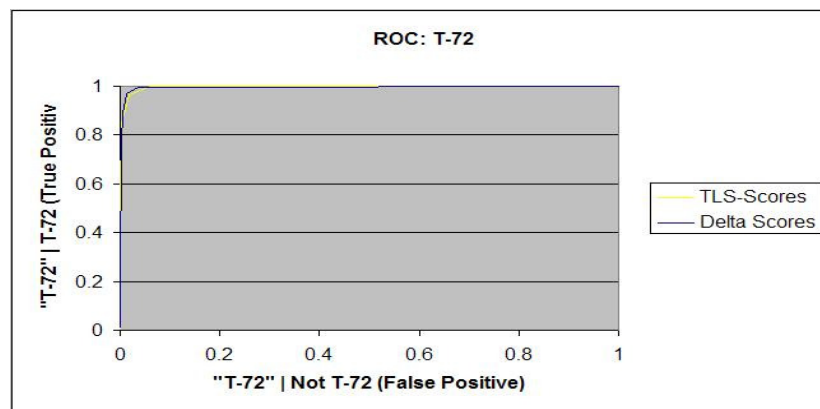


Figure 44. Normalized ROC Curve for Target 4, T-72.

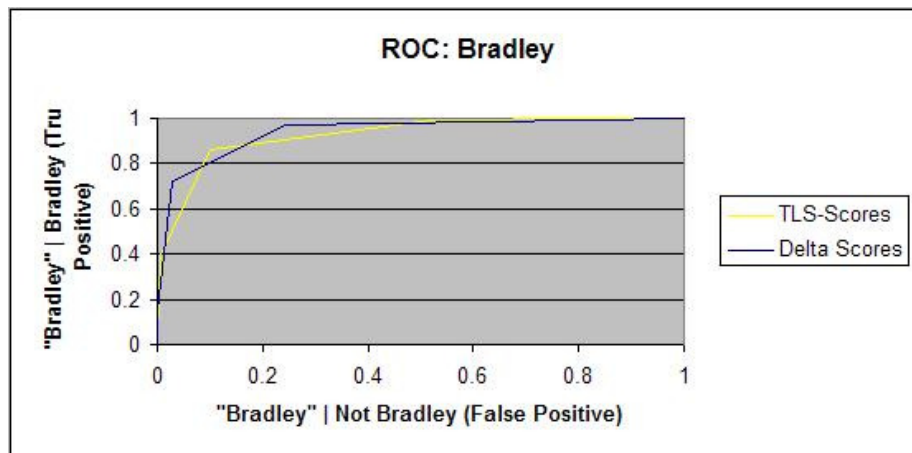


Figure 45. Normalized ROC Curve for Target 5, Bradley.

The Normalized ROC curves for the T-72 looks very good for both TLS and Delta scores. All curves show that TLS and Delta scores are providing a comparable amount of useful information. No current operating condition can be shown on these curves, as it is not possible to create one set threshold on score for any particular algorithm.

Priors are used for sensitivity analysis. Priors have some effect on Horizontal MOPS: True Positives and False Positives. However, they have a much more prominent effect on Vertical MOPS: Positive Predictive Values and Negative Predictive Values. As the maximum, minimum, and most likely number of units for each target on the field is provided, an analysis on Vertical MOPS can be accomplished on each target for each algorithm. A look at TLS-Bound's results for each target can show what information can be obtained with these graphs. The TLS-Bound results for each target are given in Figure 46 to Figure 49.

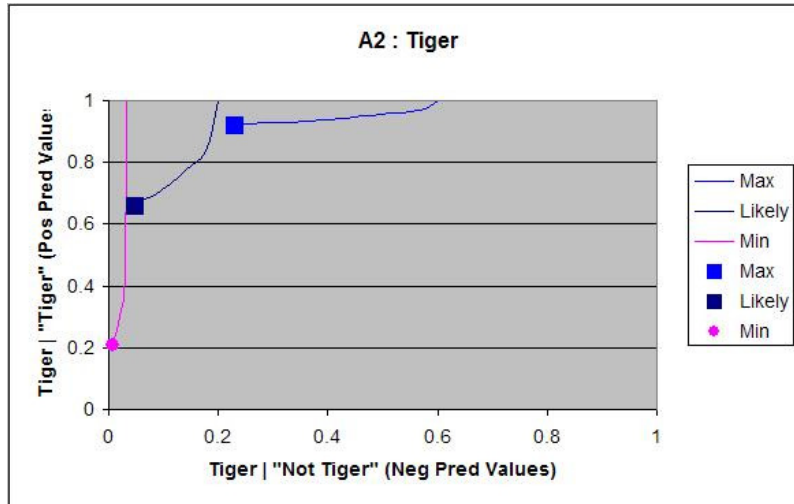


Figure 46. Priors Curve for T2, Tiger, and A2, TLS-Bound.

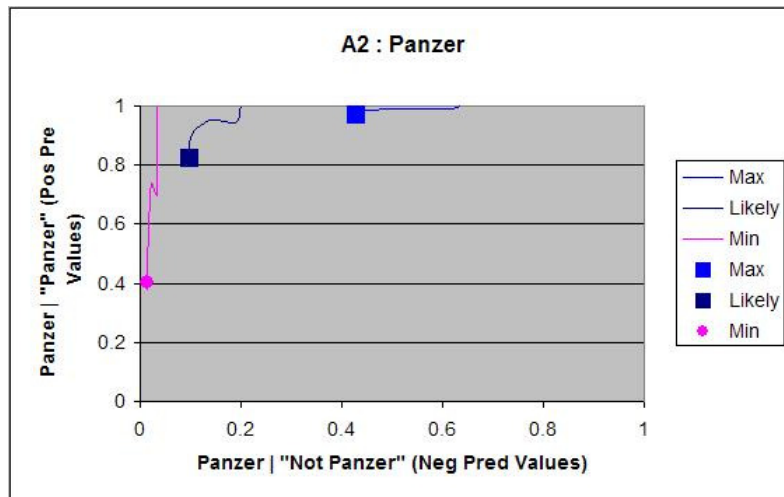


Figure 47. Priors Curve for T3, Panzer, and A2, TLS-Bound.

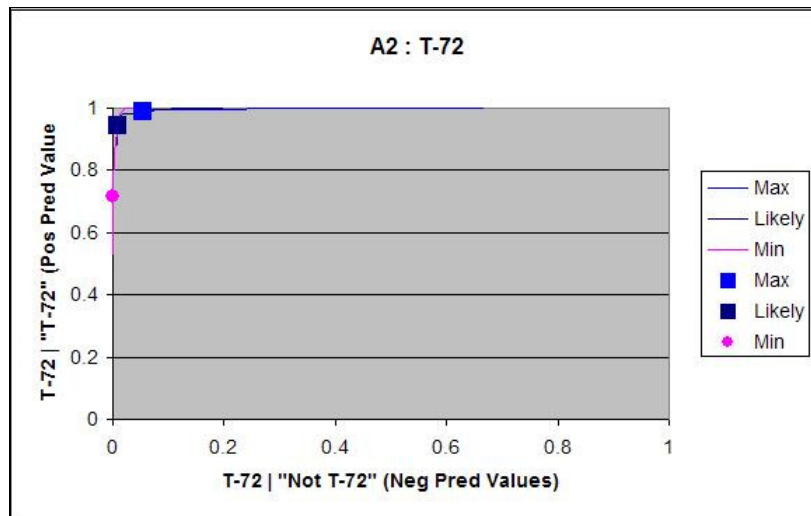


Figure 48. Priors Curve for T4, T-72, and A2, TLS-Bound.

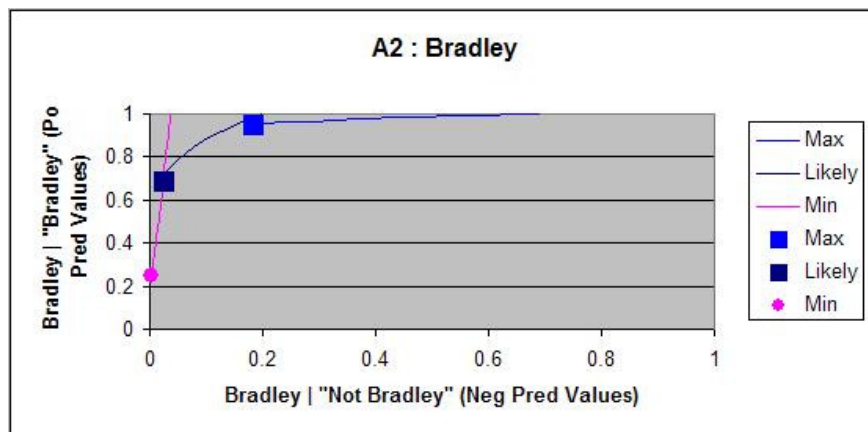


Figure 49. Priors Curve for T5, Bradley, and A2, TLS-Bound.

As noted before, the current operating conditions indicated by the points are at the bottom of each curve. This is due to TLS thresholds being set low. The minimum curve for each graph shows what the Vertical ROC curve would look like if the target's population is at the lowest possible value, while the population for each of the other targets is at their highest possible value. The maximum curve is constructed similarly.

This shows the T-72 performs well if the population is at or above the most likely level. The other targets are very sensitive to population changes. In all other cases, if the population is smaller than expected, the TLS-threshold for that target should be increased, since the curves at the minimum populations are practically vertical. A vertical line means an increased Positive Predictive Value can be obtained with very little increase in Negative Predictive Value. However, if the population might be larger than suspected, a low threshold is definitely better as most of these lines are horizontal. Bradley and Panzer show an equal tradeoff between Positive Predictive Value and Negative Predictive Value, meaning an increase in TLS-threshold will increase one value at an equal cost to the other. However, the Panzer's curve at the most likely value starts vertically. This means, the TLS-threshold may be increased, improving Positive Predictive Value at no cost to Negative Predictive Value to a point. Since the UIT is dynamic, the threshold can be adjusted until that point is reached.

The reason behind these curves is as the population decreases, the probability the target is actually TRUE given a "FALSE" indication decreases. Likewise, at any given threshold the probability the target is actually TRUE given a "TRUE" response decreases as the probability of TRUE decreases with a reduced prior. In other words, with a reduced prior, less of these targets can be encountered in the field, so the probability of encountering one of these targets decrease. However, for the T-72 the probability a target is TRUE given a "TRUE" response can still remain high if the threshold is increased.

As to changes in the operational environment, the results show the Positive Predictive and Negative Predictive Values are highly sensitive to priors, and thresholds should definitely be raised if the prior population of a target is expected to have been

reduced. On the other hand, an increased threshold when the priors have increased would result in more TRUE targets being returned as “Unknown” or another response, so the threshold should be reduced to its minimum if the prior population is expected to be high. Thus, as the situation on the battlefield changes, thresholds should definitely be changed to ensure prior predictive and negative predictive values remain good.

These results are consistent with each algorithm.

A quick glance at the Classifier Evaluation Quantifiers is very telling. The summary statistics being used are Precision, Recall, F-Value, Accuracy and Mutual Information. The summary statistics are shown in Table 29.

Table 29. Summary Statistics for first analysis.

Stats	A1	A2	A3	A4	A5
Recall	3.545987	3.715924	3.027143	2.944967	2.992978
Accuracy	0.709197	0.743185	0.605429	0.588993	0.598596
Precision	3.497836	3.733636	3.800473	3.453918	3.515746
FScore	3.424939	3.660544	3.04015	2.898849	2.94462
M Info	0.344712	0.369084	0.283576	0.245011	0.250901

Summary statistics are covered in Sections I, II and III in areas dealing with summary data, summary statistics and possible classifier quantifiers. Since these statistics are covered in three different sections, they will only be briefly covered here. Accuracy can be defined as the combined probability of True Positives and True Negatives. Precision is another way to measure Vertical MOPs. Recall is another way to measure Horizontal MOPs. An F-Score is a “harmonic mean” between the two whereas mutual information, “is defined as the Kullback-Leibler divergence between the joint distribution of <output> y and <truth> t and the product of their marginals:” (Wallach, 2004)

Higher thresholds results in better predictive values, so precision should increase from Biggest-TLS to TLS-Bound to TLS & D as more criteria are added to change a response from a declaration to “Unknown.” In the same manner, precision should increase from Big-D to D-Bound. This occurs when “Unknown” cases are treated by taking them out of the picture. Thus, TLS & D (A3) gives the greatest Precision, or Vertical MOPs. Based on Recall, or Horizontal MOPs, TLS-Bound (A2) is the best. Based on Accuracy, F-Score and Mutual Information, TLS-Bound should be used. With four classifier evaluation quantifiers to one, TLS-Bound seems to be the winner. Although it did not beat TLS & D at Precision, TLS-Bound came in second, so without adjusting thresholds, TLS-Bound should be used.

Second Analysis

Since the first analysis was accomplished with all thresholds at their minimums, the second analysis increased most thresholds to an even level to give the Delta-based algorithms a better opportunity. T-72’s TLS-threshold was not increased much due to the excellent results all-around for T-72. Tiger’s TLS-threshold was not increased very much, since the Vertical ROC curve suggests poorer performance if it is increased. Raising thresholds will result in more returns of “Unknown” and less declarations of other targets. However, it will also change Delta scores for every target.

As a reminder, one assumption stated is that, “Unknowns” are counted as “Clutter.” Clutter is not examined in ROC and Vertical ROC curves, so “Unknown” can be considered as “FALSE” in generating these graphs. Thus, the True Positive values do not show an increase, and the ROC curves show minor changes. This is because, returns as incorrect targets are becoming returns as “Unknown” and both are considered as misidentifications. In other words, “FALSE” returns are changing to

alternative “FALSE” returns as the threshold increases. The ROC curve will not change much in True Positive value for TLS-Score until the threshold increases to values that start changing “TRUE” declarations into “Unknown.”

Since True Positive cannot improve by changing thresholds for TLS-Score, future studies might be encouraged to not consider “Unknowns” in the ROC curve generation. However, considering “Unknowns” as misidentifications allow “Unknowns” to be counted and penalized against the algorithm. Furthermore, thresholds can be changed until the ROC curve begins to change for the worse. Although this will not cause an improvement in the ROC curve, it will show changes in the Vertical ROC curves. Thus, one useful technique is to change thresholds to generate the best vertical MOPs possible until a significant change in horizontal MOPs occurs.

The ROC curve can still change for Delta-based algorithms and for False Positives. Delta scores are heavily based on thresholds, so many of the returns might change as the thresholds change. The ROC curves show results for Tiger improving and Bradley getting worse in Delta-based algorithms. In addition, T-72 and Bradley both improve in the Delta-based algorithms. Bradley improves from approximately a 0.05 and 0.2 True Positive rate to a 0.63 True Positive for the Delta-based algorithms. Other changes are relatively minimal. The changes in ROC curves are provided in Figures 50 to Figure 54.

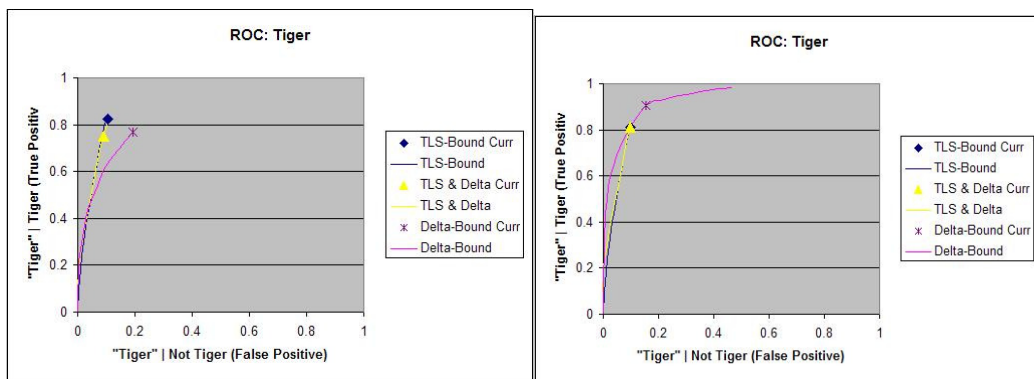


Figure 50. ROC Curve changes for Target 2, Tiger.

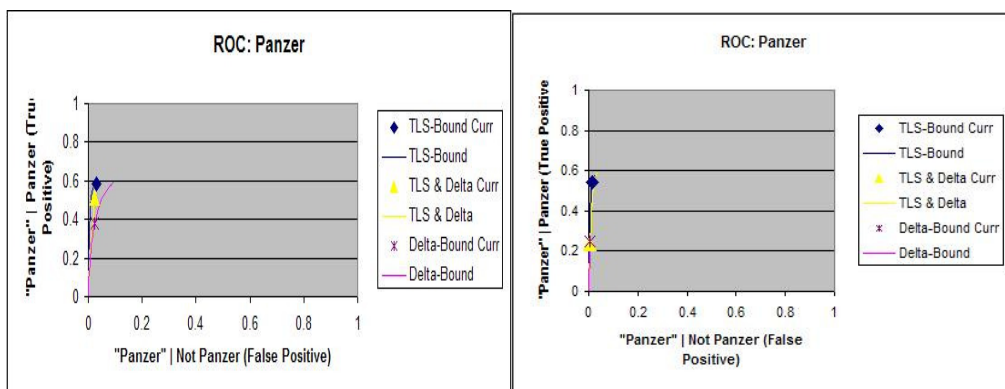


Figure 51. ROC Curve for Target 3, Panzer.

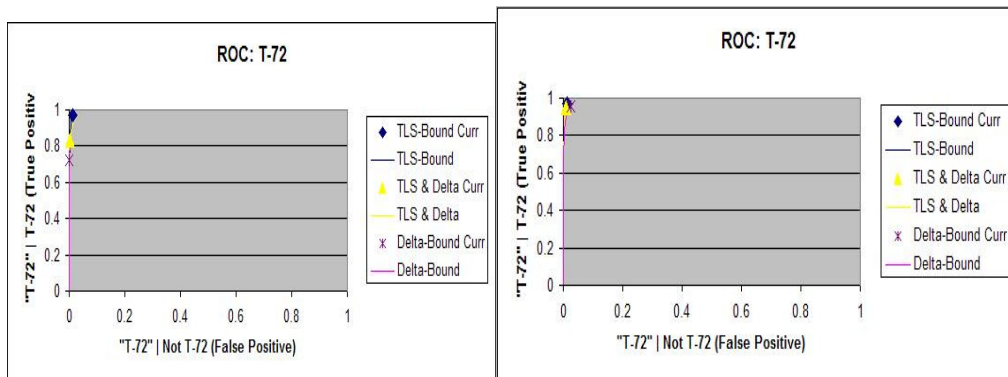


Figure 52. ROC Curve Changes for Target 4, T-72.

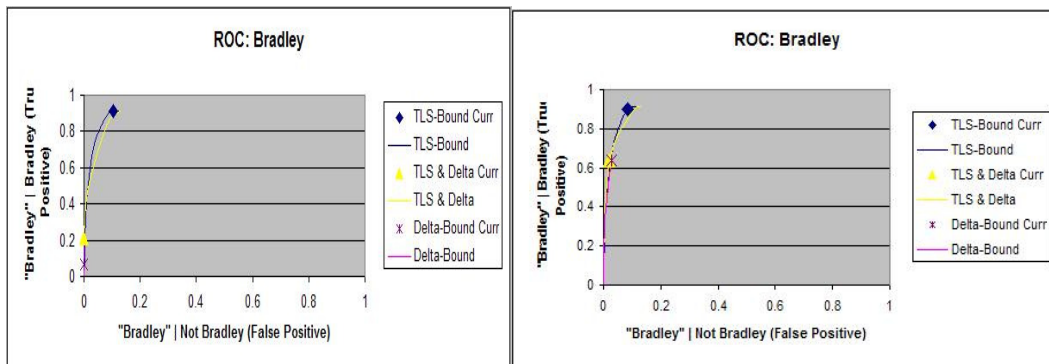


Figure 53. ROC Curve Changes for Target 5, Bradley.

Changing thresholds will change the Vertical ROC curves. A quick look shows T-72 results remains good, so it will not be further reviewed. The Vertical ROC curves for the other targets will be examined target by target starting with Tiger. The original Vertical ROC curve is given as Figure 54, while the new Vertical ROC curve is given as Figure 55.

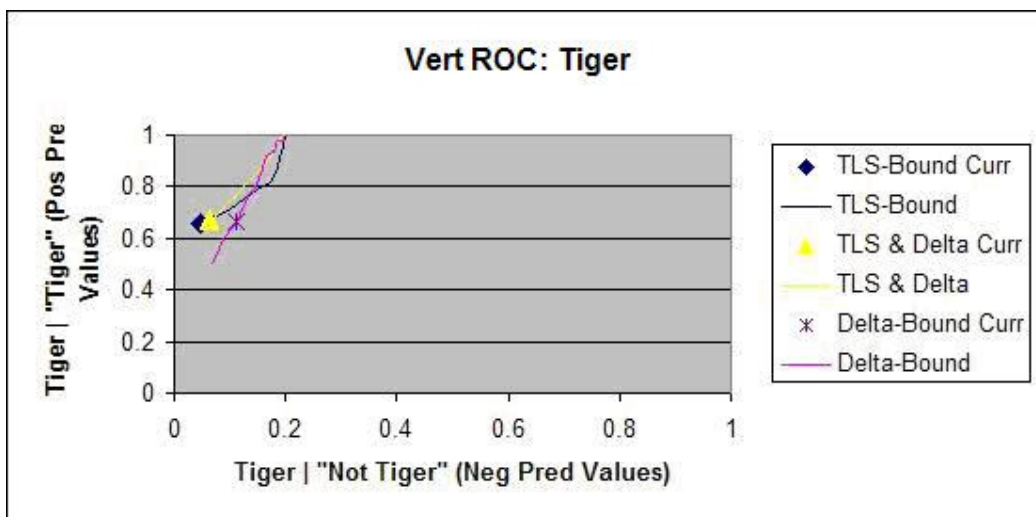


Figure 54. First Analysis ROC Curve for Target 2, Tiger.

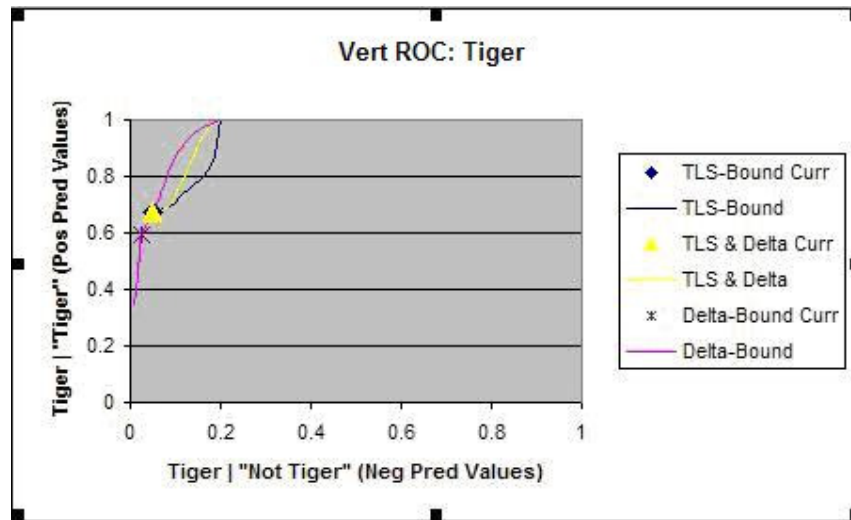


Figure 55. Second Analysis ROC Curve for Target 2, Tiger.

Only the D-Bound algorithm seemed to change significantly for Tiger as the TLS-threshold was not adjusted as much as the others. The results for the next target, Panzer, are shown in Figure 53 and Figure 54.

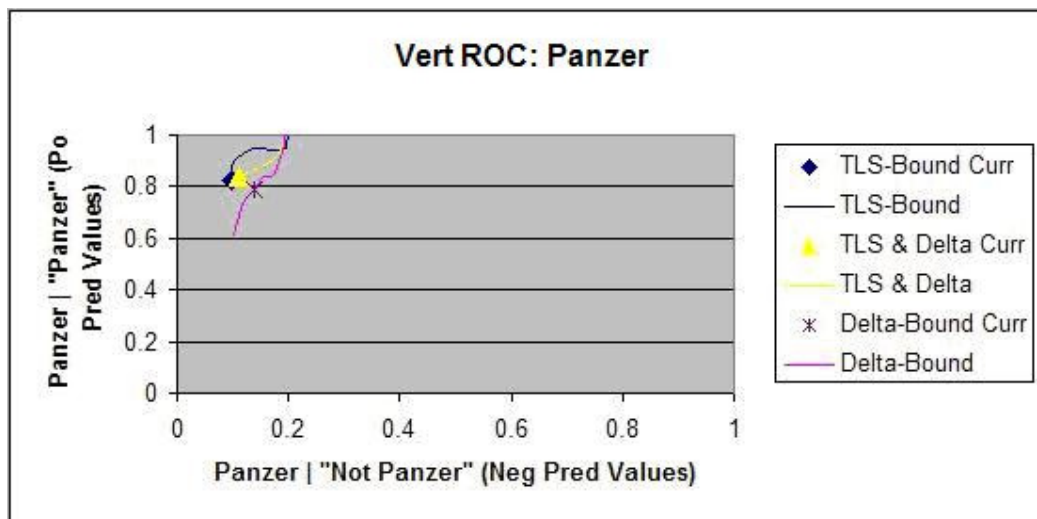


Figure 56. First Analysis Vertical ROC Curve for Target 3, Panzer.

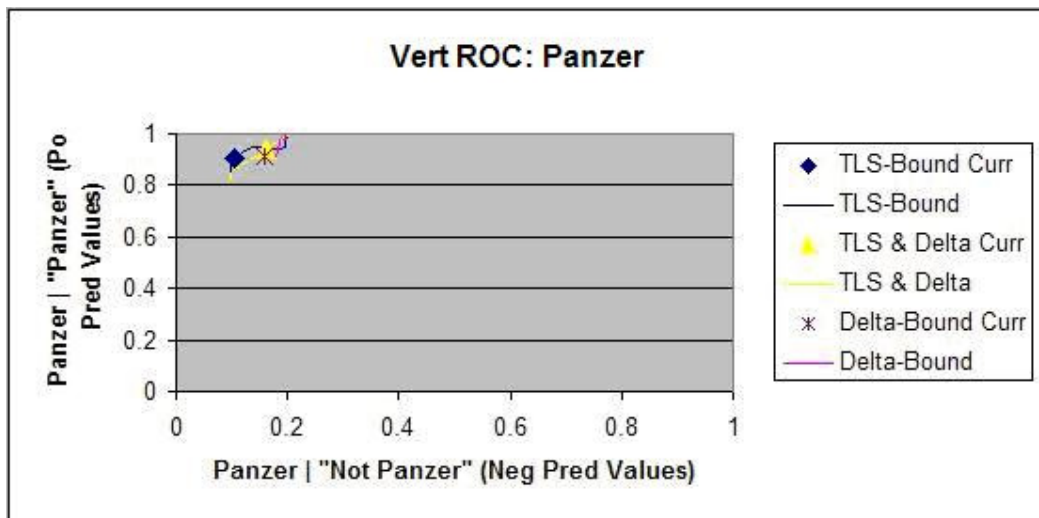


Figure 57. Vertical Second Analysis ROC Curve for Target 3, Panzer.

The Positive Predictive Values for TLS-Bound increase by 0.08, for TLS & D increase 0.1, and for D-Bound by 0.12. The Negative Predictive Values increased by 0.007, 0.05, and 0.02. Thus, TLS-Bound and D-Bound showed a good Positive Predictive improvement at a low cost of Negative Predictive Value. The results for the next target, Bradley, are shown in Figure 55 and Figure 56.

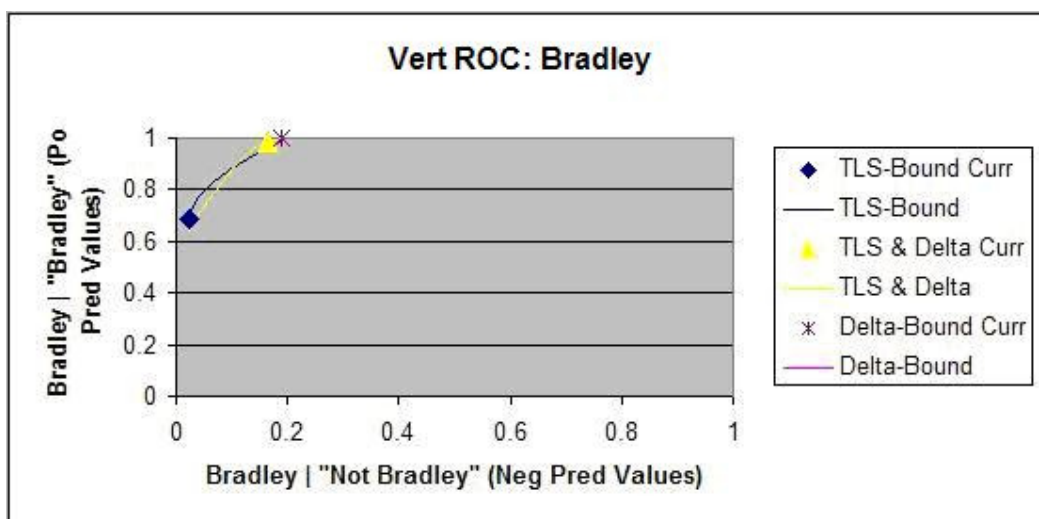


Figure 58. First Analysis Vertical ROC Curve for Target 5, Bradley.

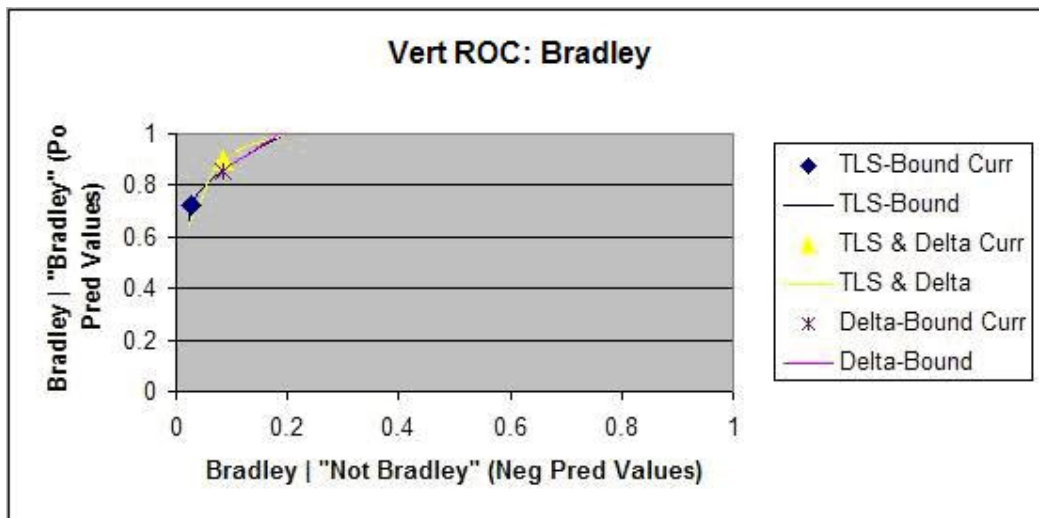


Figure 59. Vertical Second Analysis ROC Curve for Target 5, Bradley.

The Vertical ROC curves for the Bradley show only shifts for each algorithm. TLS-Bound increased its Positive Predictive Value by 0.04 at a cost of 0.0025 Negative Predictive Value. The other algorithms had equal tradeoffs.

To see how the Vertical ROC curves would be effected by populations, a look at the original and new graphs can be accomplished. Two examples are provided below to examine these effects. The first example used the Vertical ROC curves for Panzer using the D-Bound algorithm with the graphs shown in Figure 57.

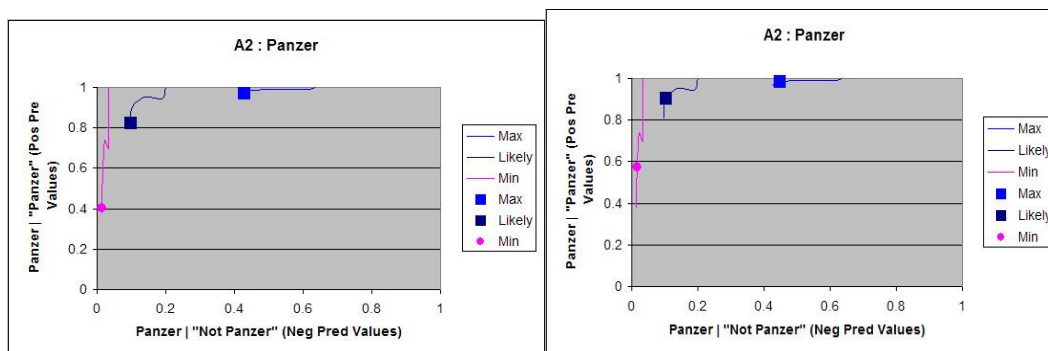


Figure 60. Vertical ROC Curve for Panzer and D-Bound with New to the Right.

A quick glance at these graphs shows that the curves for Panzer are now at higher Positive Predictive Values. The worst case scenario for the Positive Predictive Value at the current settings increased from 0.4 originally to 0.6, showing a vast improvement. The worst case Negative Predictive Value increased by 0.02, but so did its Positive Predictive Value. The most likely case improved its Positive Predictive Value by 0.078 while only worsening by .007 in Negative Predictive Value. As the thresholds have been increased, the current operating conditions are now more towards the middle of each curve.

The Vertical ROC curves only show subtle changes. Looking at the tables, it appears thresholds higher than 0.7 would start making more monumental changes in the curves. However, a quick look at the Classifier Evaluation Quantifiers can make sure the analysis is going in the right direction. The old quantifiers are given in Table 30 and the new quantifiers are given in Table 31.

Table 30. First Analysis summary statistics.

Stats	A1	A2	A3	A4	A5
Recall	3.545987	3.715924	3.027143	2.944967	2.992978
Accuracy	0.709197	0.743185	0.605429	0.588993	0.598596
Precision	3.497836	3.733636	3.800473	3.453918	3.515746
FScore	3.424939	3.660544	3.04015	2.898849	2.94462
M Info	0.344712	0.369084	0.283576	0.245011	0.250901

Table 31. Second Analysis summary statistics.

Stats	A1	A2	A3	A4	A5
Recall	3.545987	3.780227	3.387023	2.931092	3.198646
Accuracy	0.709197	0.756045	0.677405	0.586218	0.639729
Precision	3.497836	3.870468	3.873095	3.64696	3.669249
FScore	3.424939	3.765731	3.375278	2.755169	3.187094
M Info	0.344712	0.373535	0.329675	0.27734	0.30107

The tables show no change in Biggest-TLS (A1) as it cannot change with thresholds. However, TLS-Bound (A2), TLS & D (A3) and D-Bound (A5) improve in all quantifiers. Big-D (A4) improves in Precision and Mutual Information, but the other statistics get worse. With the total improvement of three algorithms, it appears the analysis is going in the right direction.

Conclusion

At minimum thresholds, TLS-Bound outperformed the other algorithms. Biggest-TLS is a special case of TLS-Bound with a zero threshold. The difference in the minimum threshold case is with TLS-Bound, scores equal to the threshold would be considered “Unknown” while Biggest-TLS would accept it as “TRUE.” However, the ROC curves show the current threshold levels generally provided an equivalent True Positive probability with a little smaller False Positive probability, which is desired. The Delta-based algorithms could improve if the Delta thresholds were decreased, but generally, TLS-Bound was the best algorithm after the first analysis.

During the second analysis, small improvements occurred with the Vertical ROC curves. The ROC curve showed improvements for most targets in all algorithms, with the biggest improvements on the Delta-based algorithms. Still, TLS-Bound seemed to dominate the Classifier Evaluation Quantifiers with either the best or close to the best scores. The best algorithm is left to the decision-maker as the benefits of Positive Predictive Probability versus Negative Predictive Probability as well as tradeoffs between all quantifiers must be weighed.

Limitations in using the original ROC curve for TLS-based algorithms are created by treating “Unknown” as “FALSE.” With higher thresholds, misidentifications turn to “Unknown” but are still counted against the True Positive.

As the threshold increases higher, correct identifications turn to “Unknown” and the ROC curve gets worse. This allows analyses to determine how much the thresholds can change to improve the Vertical ROC curves, and thus the Vertical MOPs, without seriously degrading the ROC curve, or horizontal MOPs, for the TLS-based algorithms.

Since incorrect declarations may be considered worse than “Unknown” responses, especially when a Friend is declared as “Foe” or a Foe declared as a “Friend,” future tools may wish to analyze the situation where “Unknowns” are taken out of the True Positive consideration and thus out of ROC curve generation.

By looking at all quantifiers, an algorithm, TLS-Bound, appears to be the best for this data set after two analyses. Furthermore, the results for TLS-Bound were improved by changing the user-defined parameters and studying the results. The results for “TLS & D” improved and might compete with TLS-Bound after further customization of user-defined parameters. Also, sensitivity analysis was conducted to show how changes in population could affect the results. In retrospect, populations with less variance might be more realistic and provide for better bounds on sensitivity. Still, this study shows how an algorithm can be selected along with user-defined parameters being suggested after just two analyses.

V. Discussion

Limitations

This study uses MS Excel due to availability. Unfortunately, MS Excel has limitations that may be encountered if the UIT is used in practice.

With a data file of five targets, the spawned MS Excel Workbook used a large amount of memory and space. The file generated is 210 MB. In addition, one of the worksheets used 100 of the 256 columns available. If the number of targets grows, the limit on the number of columns may be exceeded. If the number of experimental runs grows, a size limitation might be reached.

With only five targets, the MS Excel being used has shown a problem with trying to close out the workbook and often gives an error. Fortunately, having the master worksheet spawn a separate worksheet reduces the chances of corrupting the original file.

In addition, having all the files dynamic can cause problems running the macro on a slow machine. With five targets and over three thousand rows, generating the workbook can take a computer fifteen minutes. This time could increase linearly or exponentially with more data, as no study on time to compute has been attempted yet.

Future Studies

The UIT leaves much room for future study. As a first attempt to generate a tool, the UIT is used to explore possible quantifiers. For this purpose, a small number of targets were used. Future studies could include changing assumptions, treating confusers as individual confusers, using changes in ROC Curves as a bound for

changes, applying fusion, using priors differently for sensitivity analysis and testing the UIT against more data samples.

Confusers as “Unknown” rather than natural “Clutter”

Currently, confusers are being pooled together as Clutter. T1 is being counted as a correct classification for confusers. Confusers are equally sharing parts in all T1 calculations. However, confusers are different and in future studies should be considered individually.

One reason for them to be treated individually is, as the Cost Matrix in Figure 29 showed, different confusers are confused with different In-Library target. Con11 and Con7 are codes used to identify two different confusers. Con11 is most often declared a “T-72,” while Con7 is most often declared “Panzer.” Thus, the probability of any Clutter, the combined confuser category, being declared as “T-72” will depend on how many Con11s and Con7s will be encountered in the operating environment.

Furthermore, simply finding Con11 is heavily confused with T-72 may give enough reason to make adjustments to the identification system or library. Con11 could be added as an In-Library target if it is the Clutter causing the most error.

“T1” or “Clutter” being considered a correct response for any confuser is also arguable. “T1” is supposed to deal with natural clutter. Confusers can be any item, natural or man-made, that may be confused with a target. Identifying all confusers as “Clutter” assumes all confusers can be and have been identified. A better choice for a correct response of confusers is as “Unknown” rather than “Clutter.”

Relevance of Unknown Misidentification Assumption

The results from the further analysis show a different picture of the relationship between Horizontal and Vertical MOPs to user-defined parameters of thresholds for the

TLS-based algorithms. This can be explained in the handling of “Unknown” declarations.

In the assumptions, “Unknown” is considered the same as “Clutter.” Alternatively, “Unknown” can be considered a “Non-declaration.” In the second case, declaring one In-Library target as another target, for example a Tiger as a “Panzer,” is considered worse than declaring the first target, Tiger in this example, as “Unknown.” However, the UIT considers declaring a Tiger as an “Unknown” just as bad as declaring it a “Panzer.” Thus, as thresholds are increased, misidentifications as another target, like “Panzer,” are converted into “Unknown,” an equivalent misidentification.

Since a misidentification is still considered a misidentification, the Horizontal MOPs do not change at low thresholds. Once the threshold increases high enough to begin converting correct identifications into misidentifications, the Horizontal MOPs and resulting ROC curve begins to change for the worse. Although the UIT does not distinguish between “Unknown” misidentifications and misidentifications with other targets, the UIT can still be useful in finding the cutoff point where correct identifications begin to become “Unknown.”

This is useful because although the Horizontal MOPs did not change, the Vertical MOPs changed with each increase in threshold. By increasing the thresholds until a negative impact is observed on the ROC curve, a range of Vertical MOP values is found. By using this method, the most desirable Vertical MOP can be obtained without changing the Horizontal MOP. Thus, an optimal threshold level and Vertical MOP can be discovered by adjusting thresholds until the Horizontal MOP is negatively impacted.

Use Solver to generate User-Defined parameters.

A careful selection of User-Defined parameters can improve all algorithms. The selection of the User-Defined parameters, TLS-thresholds and Delta-threshold, were generated for the sample data by looking at the data. Alternatively, once a measure or quantifier is decided upon, some means of Linear Programming could be used to maximize or minimize that measure or quantifier by adjusting the User-Defined parameters to a small number of constraints. MS Excel already includes a Linear Programming method called Solver that changes values subject to constraints to maximize or minimize the value of any given cell in the workbook.

All elements of a Linear Programming problem are available. TLS-thresholds must remain between zero and one. Delta-thresholds must remain between negative one and positive one. The generated MS Excel workbooks use references. Thus, any measure or quantifier, for example Accuracy of the TLS-Bound algorithm, can be used as the desired output. Solver can seek to maximize Accuracy of the TLS-Bound algorithm by changing TLS-thresholds and Delta-thresholds subject to their constraints. Rather than Accuracy, a formula weighing different quantifiers could be entered into a cell. Nothing irrevocable can be caused by changing the workbook, since the workbook being used is generated by the Master workbook. If the workbook is damaged by accident, the Master workbook can be used to simply generate a replacement. Thus, rather than looking at the data, the initial analysis could use Solver to suggest a starting point, or any analysis could use Solver to improve an existing measure or quantifier.

Solver also requires a seed to start the Linear Programming problem. Based on the seeds, Solver could find local minimums or local maximums by mistake.

Rather than providing an in-depth look at Linear Programming at this stage of the study, Linear Programming will be suggested as a possible means for a way forward, and VBA as a possible means to generate seeds for use by the UIT.

Other Future Research.

Other research could include fusion techniques, changes in sensitivity analysis, or simply testing against more data. This thesis has shown that T-72 performs admirably in just about every case. Figure 44 even shows that normalized scores for T-72 are excellent for making a “T-72” declaration if it was practical to use these scores. A fusion technique could possibly check for whether or not the data suggest “T-72” be returned before any other consideration is made. Although the data started with only five values for each run, more information can be generated from these values, and the application of fusion techniques to this information should be considered in future research.

The method for dealing with sensitivity analysis may be too impractical. Although straight-forward and easy to understand, using priors at the absolute minimum and maximum estimates may create a wide and unreasonable bound for the Vertical ROC curves. The decision-maker might be more interested in confidence rather than the extreme what-if scenarios. Using random sampling with priors, a triangular distribution, or some number other than the extremes may be more meaningful to the decision-maker.

Contributions/Implications

The UIT adds much to selecting the best ATR algorithm. Not only does it provide a way to visually compare multiple measures for multiple declarations, it provides a method for improving each algorithm by choosing better User-Defined

parameters. Thus, algorithms are not only being compared, but customized and improved.

By creating a separate generated workbook using references to make all calculations, the UIT opens the door for Linear Programming to be used. Any user can select one measure as the primary measure or multiple quantifiers and weigh their values into one cell. MS Excel's Solver or any other Linear Programming method can change the User-Defined Parameters of TLS-threshold between zero and one and Delta-threshold between negative one and one to optimize the desired measure or quantifier.

The UIT also reduces a study's dependency on knowing User-Defined parameters beforehand. Rather than entering low, medium and high levels for each threshold or carefully selecting thresholds before conducting a study, thresholds can be changed on the spot with all quantifiers being updated as soon as the threshold changes. This allows thresholds to be adjusted and customized on the spot for desired results.

The UIT also provides a new method for viewing error with Vertical ROC curves. Vertical ROC curves provide a way to measure Vertical MOPs visually. They suggest how changing User-Defined parameters can change the effectiveness to the user. They also show when a benefit can be made in increasing Positive Predictive Value or reducing Negative Predictive Value with little cost. The Vertical ROC curve shows a tradeoff that is unseen by simply viewing Vertical MOP statistics.

The UIT also allows sensitivity analysis to be accomplished through Priors curves. By plotting Vertical ROC curves at the extreme bounds of Prior probabilities, the user can quickly identify how robust an algorithm is to uncertainties in the operational environment. It also suggests how the algorithms should or need to change as the operational environment changes.

The UIT allows multiple quantifiers to be compared and multiple errors to be analyzed in order to select the best ATR algorithm. It allows improvements to the User-Defined parameters to be measured on the spot. It allows tradeoffs in both Horizontal and Vertical MOPs to be visually seen. It allows sensitivity to priors to be observed. It opens the door for any of multiple quantifiers to be optimized. With these capabilities, the ATR UIT provides information to the decision-maker to choose the best ATR algorithm for any specified operational environment.

Appendix A: Worksheet List

Master Workbook

- “Main” – Used to browse for data
- “Thresholds”
 - Allows user to enter User-Defined parameters
 - TLS-thresholds
 - Delta-thresholds
 - Prior probabilities
 - Spawns “Generated Workbook”
- “Data” – Holds data browsed for in main

Generated Workbook

- “Data” – Holds sorted data
- “A2Curr” – Calculates current TLS-Bound operating point
- “A2” – Calculates ROC and Alternate ROC results for TLS-Bound
- “A3Curr” – Calculates current TLS & D operating point
- “A3” – Calculates ROC and Alternate ROC results for TLS & D
- “A5Curr” – Calculates current D-Bound operating point
- “A5” – Calculates ROC and Alternate ROC results for D-Bound
- “Algorithms”
 - Has table and graphs for ROC Curves
 - Has table and graphs for Alternate ROC Curves
 - Allows user to change User-Defined Parameters

- References data from “A2Curr,” “A3Curr,” “A5Curr,” “A2,” “A3” and “A5”
- “Scores” – Calculates ROC and Alternate ROC results for TLS
- “Scores2” – Calculates ROC and Alternate ROC results for Delta Scores
- “Algorithms2”
 - Has table and graphs for Measures ROC Curves
 - References data from “Scores” and “Scores2”
- “PriorsA2” – Generates Priors Curve for TLS-Bound
- “PriorsA3” – Generates Priors Curve for TLS & D
- “PriorsA5” – Generates Priors Curve for D-Bound
- “Stats” – Calculates Classifier Evaluation Metrics
- “ClassMetrics” – Presents Classifier Evaluation Metrics
- “Sheet1” possibly through “Sheet3”
 - Left over sheets from the creation of the generated workbook
 - Not deleted automatically as the creation of these sheets depend on the computer user’s MS Excel preferences.

Appendix B: Pseudocode

Public Variables

```
Public Const ThisXLFile As String = "<Name of Master Workbook>.xls"  
Public MyMasterBook As Workbook 'The Master Workbook Referenced twice.  
Public SummarySheet As Worksheet 'Contains copied data.
```

Function MyParseCol(MyCol As Range) As String

Returns the Column of a cell with no references.

Sub MyGetPath2()

Allows the user to browse for a data file.
Records the filename and the directory path to the data file.
Calls MyReturnPath() to determine the directory path.
Calls MyReturnFilename() to determine the filename.
Calls MyCopyData() and MyThresholdData().

Function MyReturnPath(LongThing As String) As String

Determines the Directory Path of the browsed file.

Function MyReturnFileName(LongThing As String) As String

Determines the filename of the returned file.

Sub MyCopyData(MyPath As String, MyFileName As String)

This sub creates a copy into a new file of the data in the data file. It adds the copied data into a worksheet on the new file entitled "data." It modifies "data" and "thresholds"

Assign the workbook directory to the current directory.
Sets two of the public variables

Set MyMasterBook = ThisWorkbook
 Set SummarySheet = Workbooks(ThisXLFile).Worksheets("Data")
 Clears the data sheet in the master file.
 Copies information.
 Pastes information.
 Autofits

Sub MyThresholdData()

Copies information into the master file "Thresholds" worksheet.
 Adjusts colors and borders accordingly.

Private Sub MyCreateOutput(MyOutFile As String)

Spawns a separate output file.
 Adds all relevant worksheets to the new file along with the correct format. The worksheets added to the new, output file are: "ClassMetrics," "Stats," "PriorsA5," "PriorsA3," "PriorsA2," "Algorithms2," "Scores2," "Scores," "Algorithms," "A5," "A5Curr," "A3," "A3Curr," "A2," "A2Curr," and "Data."

Sub MyComputeAlgorithms()

Calls all remaining algorithms once the user defined parameters have been entered. Calculations are turned on and off to shorten computation time. Entire Sub program is included below:

Call MyCreateOutput(MyOutFile)
 Call MyAlgorithms(MyOutFile)
 Call MyDataSheet(MyOutFile)

Application.Calculation = xlManual

Call MyA2Curr(MyOutFile)
 Call MyA2(MyOutFile)
 Call MyA3Curr(MyOutFile)
 Call MyA3(MyOutFile)

Call MyA5Curr(MyOutFile)

Call MyA5(MyOutFile)

Calculate

Application.Calculation = xlAutomatic

Call MyAlgoFill(MyOutFile)

Call MyAlgoGraphs(MyOutFile)

Call MyScores(MyOutFile)

Call MyScores2(MyOutFile)

Call MyAlgorithms2(MyOutFile)

Call MyScoreGraphs(MyOutFile)

Call MyPriors(MyOutFile, "PriorsA2")

Call MyPriors(MyOutFile, "PriorsA3")

Call MyPriors(MyOutFile, "PriorsA5")

Call MyPriorsA2(MyOutFile, "PriorsA2", "A2", "A2Curr")

Call MyPriorsA2(MyOutFile, "PriorsA3", "A3", "A3Curr")

Call MyPriorsA2(MyOutFile, "PriorsA5", "A5", "A5Curr")

Call MyStatsFile(MyOutFile)

Call MyFixBayes(MyOutFile, "A2Curr")

Call MyFixBayes(MyOutFile, "A2")

Call MyFixBayes(MyOutFile, "A3Curr")

Call MyFixBayes(MyOutFile, "A3")

Call MyFixBayes(MyOutFile, "A5Curr")

Call MyFixBayes(MyOutFile, "A5")

Call MyFixBayes(MyOutFile, "Scores")

Call MyFixBayes(MyOutFile, "Scores2")

Sub MyFixBayes(MyOutFile As String, MyCurFile As String)

Uses the created output filename and the worksheet in question as parameters.

Fixes six of the algorithm computation sheets by adjusting the $P(T|T)$ on each file.

Uses many references as each target must be done separately and current sheets must be updated with information already calculated

Fill in Probabilities referring to “Algorithms”

Place under each calculation.

Requires a nested, For, Do, For loop.

Inside For does each row.

Do loop does each threshold.

Outside For does each target.

Calculates each target’s $P(\text{“T”} \mid \text{false target})$ for each false target

Replaces $P(\text{“T”} \mid \sim T)$

Sub MyStatsFile(MyOutFile As String)

Fills in “Stats” and “ClassMetrics”

Separate declarations by placing Truths into the declaration column for each algorithm.

Count Truths per column of declaration

Start Calculating each summary statistic

Write priors, $P(T)$

Recall: Write $P(\text{“T”} \mid T)$

Update ClassMetrics

Accuracy: Write $P(T \ \& \ \text{“T”})$

Sum columns to find $P(\text{“T”})$

Update ClassMetrics

Precision: Write $P(T \mid \text{“T”})$

F Score: Uses Precision and Recall

Update ClassMetrics

Mutual Information:

Write $P(T) * P(\text{“T”})$

Write Quotient: $P(T \ \& \ \text{“T”}) / (P(T) * P(\text{“T”}))$

Write Log: $\text{Log}(\text{Above})$

Write Product: $P(T) * P(\text{“T”}) * \text{Above}$

Sum all for MI

Sub MyMakePriorGraph(...)

Make three lines (Max, Most Likely, Min)

Make three current points

Add descriptive chart options

Sub MyPriorsA2(MyOutFile As String, MyPlaceSheet As String, MyFirstSheet As String, MySecondSheet As String)

Follows MyPriors() which writes the Full Priors Matrix references

Writes Thresholds

Fills in the top heading from either A2, A3 or A5

Fills in the rest of the top with the new Probabilities for P(T) and P(~T)

Fills in the top heading from either A2Curr, A3Curr or A5Curr

Fills in the rest of the top with the new Probabilities for P(T) and P(~T)

Copy information from Top into a table at the Bottom

Sends information to make a graph

Call MyMakePriorGraph(MyOutFile, MyPlaceSheet,
MyGraphCell, MyXTitle, MyYTitle, MyGTitle)

Sub MyPriors(MyOutFile As String, MyPriorFile As String)

Writes the Full Priors Matrix references from Algorithms into the Priors
worksheets

Sub MyDataSheet(MyOutFile As String)

Copies data from the master file into the output file.

Classifies the data file

Indicates if it is clutter or not

Ranks the rest by priority

Sorts the data

The sort is used in creating meaningful CountIfs to determine Statistics

Sub MyMakeEntry(MyOutFile As String, MyEntryCell As Range)

Colors the cell differently from the background

Used to determine which cells can be changed

Sub MyAlgorithms2(MyOutFile As String)

Write Heading

Write TLS-Scores

Write Alternate ROC TLS-Scores

Write Delta Scores

Write Alternate ROC Delta-Scores

Sub MyAlgorithms(MyOutFile As String)

Write Headings

Calculate Priors

Sub MyAlgoFill(MyOutFile As String)

Write Headings

Write A2Curr ROC then Alternate ROC

Write A2 ROC then Alternate ROC

Write A3Curr ROC then Alternate ROC

Write A3 ROC then Alternate ROC

Write A5Curr ROC then Alternate ROC

Write A5 ROC then Alternate ROC

Sub MyMakeScore(MyOutFile As String, MyStartCell As Range, MyXTitle As String, MyYTitle As String, MyGTitle As String)

Creates the graphs for Algorithms2

Two lines on each curve

Sub MyMakeGraph(MyOutFile As String, MyStartCell As Range, MyXTitle As String, MyYTitle As String, MyGTitle As String)

Sub MyScoreGraphs(MyOutFile As String)

Setup Graph

Call MyMakeScore() to make ROC graph

Setup next graph

Call MyMakeScore() to make Alternate ROC graph

Sub MyAlgoGraphs(MyOutFile As String)

Setup Graph

Call MyMakeGraph () to make ROC graph

Setup next graph

Call MyMakeGraph () to make Alternate ROC graph

Differs from MyScoreGraphs by the different function calls (due to different requirements), “Algorithms” vs “Algorithms2” and different size of offsets needed to traverse the page.

Sub MyScores(MyOutFile As String)

Copy Data

Reference Probabilities

Write/Calculate Scores

For each target, vary thresholds and calculate stats for ROC and Alt ROC

Curves

Sub MyScores2(MyOutFile As String)

Copy Data

Reference Probabilities

Write/Calculate Delta

Write new Deltas with zeroes instead of negatives

Write/Calculate Delta Scores

For each target, vary thresholds and calculate stats for ROC and Alt ROC

Curves

Sub MyA2(MyOutFile As String), MyA3(...), MyA5(...),

Copy Data

Calculate Max
 Calculate Delta
 Write the Max_Delta
 Write the Threshold
 Writes the Delta
 Calculate Beat_Delta: The amount the Max_Delta beats the next delta.
 Paste TLS Data
 Make room for Delta Data
 Make room for Other Delta Data
 Paste TLS Thresholds
 Paste Delta Threshold
 Calculate Delta Data: TLS Data – TLS Threshold
 Paste Probabilities
 Calculate Max: The maximum TLS Score
 Calculate A1: Max TLS Score's Target
 Calculate Other Delta Data
 Eliminate the Delta of the Max TLS Score's Target
 Write other data so it can be summed
 Other_D:
 MyA2, MyA5: Highest delta not of Max TLS Score's Target
 MyA3: Highest delta not of Max_D Score's Target
 Thres: Threshold of the Max TLS Score's Target
 Delta: of the Max TLS Score's Target
 Beat_D: Used in A6, a possible algorithm. Changed for use in myA3
 MyA2, MyA5: Max_D – Other_D (based on TLS)
 MyA3: Max_D – Other_D (based on Delta)
 Calculate A2: A1, but Delta must be >0
 MaxD: Maximum Delta
 Calculate A3:
 Delta = Max Delta
 Delta > Other_D doesn't work, because Other_D had to change
 Calculate A4: Max Delta's Target
 Calculate A5:
 Max Delta's Target
 Beat_D > D_Thres
 Correct for A2 and A5. Wrong for A3

Calculate A6: Delta > D_Thres

Write Top Heading

Write Bottom Logic

Changes from MyA2, MyA3, and MyA5

“A” = “T” and T

“B” = “T” and ~T

“C” = “~T” and T

“D” = “~T” and ~T

Code for “A,” “B,” “C,” and “D” written below

Write Thresholds

Write Top Logic

“A,” “B,” “C,” and “D” code

MyA2(), MyCurrA2()

```
MyCurString = "=IF(AND(" & MyNewPlace.Offset(0, -1 -  
MyNewCount).AddressLocal(RowAbsolute:=False, ColumnAbsolute:=True) & "=1,"  
& MyNewPlace.Offset(0, -3 - MyNewCount).AddressLocal(RowAbsolute:=False,  
ColumnAbsolute:=True) & "=1, $I23> " & MyNewPlace.Offset(-1,  
0).AddressLocal(RowAbsolute:=True, ColumnAbsolute:=False) _  
& "),""A"",IF(AND(" & MyNewPlace.Offset(0, -1 -  
MyNewCount).AddressLocal(RowAbsolute:=False, ColumnAbsolute:=True) &  
"=1,$I23>" & MyNewPlace.Offset(-1, 0).AddressLocal(RowAbsolute:=True,  
ColumnAbsolute:=False) & "," & MyNewPlace.Offset(0, -3 -  
MyNewCount).AddressLocal(RowAbsolute:=False, ColumnAbsolute:=True) _  
& "=0),""B"",IF(AND(OR(" & MyNewPlace.Offset(0, -1 -  
MyNewCount).AddressLocal(RowAbsolute:=False, ColumnAbsolute:=True) &  
"=0,$I23<" & MyNewPlace.Offset(-1, 0).AddressLocal(RowAbsolute:=True,  
ColumnAbsolute:=False) & ")," & MyNewPlace.Offset(0, -3 -  
MyNewCount).AddressLocal(RowAbsolute:=False, ColumnAbsolute:=True) _  
& "=1),""C"",IF(AND(OR(" & MyNewPlace.Offset(0, -1 -  
MyNewCount).AddressLocal(RowAbsolute:=False, ColumnAbsolute:=True) &  
"=0,$I23<" & MyNewPlace.Offset(-1, 0).AddressLocal(RowAbsolute:=True,  
ColumnAbsolute:=False) & ")," & MyNewPlace.Offset(0, -3 -  
MyNewCount).AddressLocal(RowAbsolute:=False, ColumnAbsolute:=True) &  
"=0),""D"", ""E"")))))"
```

```

MyA3(), MyCurrA3()
    MyCurString = "=IF(AND(" & MyNewPlace.Offset(0, -1 -
MyNewCount).AddressLocal(RowAbsolute:=False, ColumnAbsolute:=True) & "=1,"
& MyNewPlace.Offset(0, -3 - MyNewCount).AddressLocal(RowAbsolute:=False,
ColumnAbsolute:=True) & "=1, $I23> " & MyNewPlace.Offset(-1,
0).AddressLocal(RowAbsolute:=True, ColumnAbsolute:=False) _
    & ", $I23-" & MyNewPlace.Offset(-1,
0).AddressLocal(RowAbsolute:=True, ColumnAbsolute:=False) _
    & ">=$J23),"A",IF(AND(" & MyNewPlace.Offset(0, -1 -
MyNewCount).AddressLocal(RowAbsolute:=False, ColumnAbsolute:=True) &
"=1, $I23>" & MyNewPlace.Offset(-1, 0).AddressLocal(RowAbsolute:=True,
ColumnAbsolute:=False) _
    & ", $I23-" & MyNewPlace.Offset(-1,
0).AddressLocal(RowAbsolute:=True, ColumnAbsolute:=False) & ">=$J23," &
MyNewPlace.Offset(0, -3 - MyNewCount).AddressLocal(RowAbsolute:=False,
ColumnAbsolute:=True) _
    & "=0),"B",IF(AND(OR(" & MyNewPlace.Offset(0, -1 -
MyNewCount).AddressLocal(RowAbsolute:=False, ColumnAbsolute:=True) &
"=0, $I23<" & MyNewPlace.Offset(-1, 0).AddressLocal(RowAbsolute:=True,
ColumnAbsolute:=False) _
    & ", $I23-" & MyNewPlace.Offset(-1,
0).AddressLocal(RowAbsolute:=True, ColumnAbsolute:=False) & "<$J23)," &
MyNewPlace.Offset(0, -3 - MyNewCount).AddressLocal(RowAbsolute:=False,
ColumnAbsolute:=True) _
    & "=1),"C",IF(AND(OR(" & MyNewPlace.Offset(0, -1 -
MyNewCount).AddressLocal(RowAbsolute:=False, ColumnAbsolute:=True) &
"=0, $I23<" & MyNewPlace.Offset(-1, 0).AddressLocal(RowAbsolute:=True,
ColumnAbsolute:=False) _
    & ", $I23-" & MyNewPlace.Offset(-1,
0).AddressLocal(RowAbsolute:=True, ColumnAbsolute:=False) & "<$J23)," &
MyNewPlace.Offset(0, -3 - MyNewCount).AddressLocal(RowAbsolute:=False,
ColumnAbsolute:=True) _
    & "=0),"D","E"))))"

```

MyA5(), MyCurrA5()

```

MyCurString = "=IF(AND(" & MyNewPlace.Offset(0, -1 -
MyNewCount).AddressLocal(RowAbsolute:=False, ColumnAbsolute:=True) & "=1,"
& MyNewPlace.Offset(0, -3 - MyNewCount).AddressLocal(RowAbsolute:=False,
ColumnAbsolute:=True) & "=1, $L23>" & MyNewPlace.Offset(-1,
0).AddressLocal(RowAbsolute:=True, ColumnAbsolute:=False) _
& "),""A"",IF(AND(" & MyNewPlace.Offset(0, -1 -
MyNewCount).AddressLocal(RowAbsolute:=False, ColumnAbsolute:=True) &
"=1,$L23>" & MyNewPlace.Offset(-1, 0).AddressLocal(RowAbsolute:=True,
ColumnAbsolute:=False) & "," & MyNewPlace.Offset(0, -3 -
MyNewCount).AddressLocal(RowAbsolute:=False, ColumnAbsolute:=True) _
& "=0),""B"",IF(AND(OR(" & MyNewPlace.Offset(0, -1 -
MyNewCount).AddressLocal(RowAbsolute:=False, ColumnAbsolute:=True) &
"=0,$L23<" & MyNewPlace.Offset(-1, 0).AddressLocal(RowAbsolute:=True,
ColumnAbsolute:=False) & ")," & MyNewPlace.Offset(0, -3 -
MyNewCount).AddressLocal(RowAbsolute:=False, ColumnAbsolute:=True) _
& "=1),""C"",IF(AND(OR(" & MyNewPlace.Offset(0, -1 -
MyNewCount).AddressLocal(RowAbsolute:=False, ColumnAbsolute:=True) &
"=0,$L23<" & MyNewPlace.Offset(-1, 0).AddressLocal(RowAbsolute:=True,
ColumnAbsolute:=False) & ")," & MyNewPlace.Offset(0, -3 -
MyNewCount).AddressLocal(RowAbsolute:=False, ColumnAbsolute:=True) &
"=0),""D"",""E""))))"

```

Sub MyA2Curr(...), MyA3Curr(...), MyA5Curr(...)

Altered from MyA2(), MyA3(), and MyA5() respectively

Eliminates loops

Measures against user-defined threshold not a predetermined set

Bibliography

1. Albrecht, Timothy W. "Combat Identification with Sequential Observations, Rejection Option, and Out-Of-Library Targets," Air Force Institute of Technology (AU), Wright-Patterson AFB OH, (05-03)
2. Hopley, Lara. Schalkwyk, Jo van.
<http://www.anaesthetist.com/mnm/stats/roc/Findex.htm>
3. Leap, Nathan J., Clemans, Paul P., Bauer, Kenneth W. Jr., and Oxley, Mark E. "An Investigation of the Effects of Correlation and Autocorrelation on Classifier Fusion and Optimal Classifier Ensembles," Air Force Institute of Technology (AU), Wright-Patterson AFB OH
4. Parker, David. "FW: Re: Definition of "Score"," E-mail to Bauer, Kenneth W Ph.D. 29 Sep. 2006.
5. Roli, Fabio. "FUSION 2002 TUTORIAL: Fusion of Multiple Classifiers Part I," University of Cagliari, Dept. of Electrical and Electronics Eng., Italy, 2002.
6. Roli, Fabio. "FUSION 2002 TUTORIAL: Fusion of Multiple Classifiers Part II," University of Cagliari, Dept. of Electrical and Electronics Eng., Italy, 2002.
7. Roli, Fabio. "FUSION 2002 TUTORIAL: Fusion of Multiple Classifiers Part III," University of Cagliari, Dept. of Electrical and Electronics Eng., Italy, 2002.
8. Roli, Fabio. "FUSION 2002 TUTORIAL: Fusion of Multiple Classifiers Part IV," University of Cagliari, Dept. of Electrical and Electronics Eng., Italy, 2002.
9. Sadowski, Charles. "RE: Data for AFIT student," E-mail to Capt. David A. Kerns. 18 Aug. 2006.
10. Sadowski, Charles. "RE: Image quality," E-mail to Dr. Kenneth Bauer. 5 Jun. 2006.

11. Simon, Steve. <http://www.childrensmrcy.org/stats/ask/roc.asp>
12. Wackerly, Dennis D., William Mendenhall, and Richard L, Schaeffer.
Mathematical Statistics with Applications. Pacific Grove, CA: Duxbury, 2002.
13. Wallach, “Evaluation Metrics for Hard Classifiers,” University of Cambridge,
27 Nov 2004
14. Wikipedia, http://en.wikipedia.org/wiki/Positive_predictive_value

REPORT DOCUMENTATION PAGE				Form Approved OMB No. 074-0188	
<p>The public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of the collection of information, including suggestions for reducing this burden to Department of Defense, Washington Headquarters Services, Directorate for Information Operations and Reports (0704-0188), 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to a penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number.</p> <p>PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ADDRESS.</p>					
1. REPORT DATE (DD-MM-YYYY) 22-03-2007		2. REPORT TYPE Master's Thesis		3. DATES COVERED (From – To) Jun 2006 – Mar 2007	
4. TITLE AND SUBTITLE AUTOMATIC TARGET RECOGNITION USER INTERFACE TOOL				5a. CONTRACT NUMBER	
				5b. GRANT NUMBER	
				5c. PROGRAM ELEMENT NUMBER	
6. AUTHOR(S) Kerns, David A., Captain, USAF				5d. PROJECT NUMBER	
				5e. TASK NUMBER	
				5f. WORK UNIT NUMBER	
7. PERFORMING ORGANIZATION NAMES(S) AND ADDRESS(S) Air Force Institute of Technology Graduate School of Engineering and Management (AFIT/EN) 2950 Hobson Street, Building 642 WPAFB OH 45433-7765				8. PERFORMING ORGANIZATION REPORT NUMBER AFIT/GOR/ENS/07-15	
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) Intentionally Left Blank				10. SPONSOR/MONITOR'S ACRONYM(S)	
				11. SPONSOR/MONITOR'S REPORT NUMBER(S)	
12. DISTRIBUTION/AVAILABILITY STATEMENT APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.					
13. SUPPLEMENTARY NOTES					
14. ABSTRACT <p>A computer tool to aid in selecting the best Automatic Target Recognition (ATR) algorithm is developed. The program considers many quantifiers, accepts user-defined parameters, allows for changes in the operational environment and presents results in a meaningful way. It is written for Microsoft Excel.</p> <p>An ATR algorithm assigns a class label to a recognized target. General designations can include "Friend" and "Foe." The error of designating "Friend" as "Foe" as well as "Foe" as "Friend" comes with a high cost. Studying each algorithm's error can minimize this cost. Receiver Operating Characteristic (ROC) curves provide only information on the probabilities given a system state of declaring up to three class labels: "True," "False" or "Unknown." Other quantifiers, including an alternate ROC curve, are developed in this study to provide information on the probability of a system state given any of multiple declarations, which is more useful to the user. Sensitivity to prior probabilities, suggestions for user-defined parameters and areas for future research are identified as the User Interface Tool is described in detail in this thesis.</p>					
15. SUBJECT TERMS CID, Target Recognition, Target Classification, Error Analysis, Interfaces, Vertical, Operating Characteristic Curve, Prior Probability, Predictions					
16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Kenneth W. Bauer (ENS)
U	U	U	UU	150	19b. TELEPHONE NUMBER (Include area code) (937) 255-6565, ext 4328; e-mail: Kenneth.Bauer@afit.edu

Standard Form 298 (Rev. 8-98)
Prescribed by ANSI Std. Z39-18